3-6-2006

Evaluation of Energy Costs and Error Performance of Range-Aware Anchor-Free Localization Algorithms for Wireless Sensor Networks

Gustav Julio Jordt

Evaluation of Energy Costs and Error Performance of Range-Aware, Anchor-Free Localization Algorithms for Wireless Sensor Networks

THESIS

Gustav Julio Jordt, Captain, USAF

AFIT/GCE/ENG/06-02

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GCE/ENG/06-02

Evaluation of Energy Costs and Error Performance of Range-Aware, Anchor-Free Localization Algorithms for Wireless Sensor Networks

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Computer Engineering

Gustav Julio Jordt, B.S.E.E.C.S.

Captain, USAF

March 2006

AFIT/GCE/ENG/06-02

Evaluation of Energy Costs and Error Performance of Range-Aware, Anchor-Free Localization Algorithms for Wireless Sensor Networks

Gustav Julio Jordt, B.S.E.E.C.S.
Captain, USAF

Approved:

Dr. Rusty O. Baldwin (Chairman)	6 Mar 06 date
Dr. Barry E. Mullins (Member)	6 Mar 06 date
Dr. John F. Raquet (Member)	6 MAR 06 date

AFIT/GCE/ENG/06-02

Abstract

This research examines energy and error performance tradeoffs in Anchor-Free Range-Aware Wireless Sensor Network (WSN) Localization algorithms in different network environments. A concurrent and an incremental algorithm (Anchor Free Localization (AFL) and Map Growing) are examined under varying network sizes, densities, distribution methods, and range error conditions. Despite current expectations, even the most expensive configurations do not expend significant amounts of battery life (at most 0.4% of a typical node battery). This implies very little energy conservation is possible during localization which is contrary to significant current research that tries to provide accurate localization while conserving energy usage.

AFL is twice as accurate but uses up to 6 times more communication. AFL's position refinement phase adds 700 - 2500 messages but significantly reduces error. Map Growing averages 35 total messages. For both, node degree is the single most important factor, accounting for over 90% of the variation in communication. As node degree increases, Map Growing communication increases, while AFL transmissions drop. During refinement, nodes with more neighbors refine quicker using fewer messages. Between degree 12 and 16, many nodes receive the same message. This effect overpowers the previous resulting in more AFL received bits. Other factors have little effect on communication. Network size dramatically degrades Map Growing accuracy but has little effect on AFL. Built from simulation data, the Energy Consumption Model predicts the energy usage of incremental and concurrent algorithms used in networks with varying size, density, and deployment methods. The model is applied to current wireless sensor nodes. Among the many WSN applications, the military operations include target tracking, surveillance, and remote intelligence gathering. For military use, WSNs need to be durable, flexible, cheap, low-maintenance and long lasting. Anchor-Free and Range-Aware algorithms best fit this need.

Acknowledgements

With God all things are possible. -Matthew Ch XIX, V. 26

I offer my sincere thanks and appreciation to my adviser, Dr. Rusty O. Baldwin. His thoughtful guidance, confidence in my abilities, disciplined approach and legendary attention to detail brought out the best in me. Additionally, I thank Dr. Barry E. Mullins and Dr. John F. Raquet for their insight, useful advice, support, encouragement and expertise. Their constructive comments and probing questions ensured all issues were properly addressed.

This work would not be remotely possible without the love, support, understanding, patience, forgiveness, and sacrifice of my wonderful wife. She held everything together when I could not be there to help. Special thanks also goes to my mother in law, whose assistance was immensely and vitally helpful. I also thank my newborn daughter for inspiring me and through her innocence, joy, and growth, provided me with much needed respite and escape from the arduous work.

I thank my parents who, through great sacrifice, provided an excellent education, an encouraging spirit, and all the support I could ask for. The importance they placed on education, learning, and expanding one's mind has stayed with me my entire life.

Finally, I give thanks and glory to God. He has provided the strength, patience, and serenity to survive this process. Any accomplishment I have done is only possible by His grace.

Gustav Julio Jordt

Table of Contents

Page

Page

Page

List of Figures

List of Tables

List of Abbreviations

Abbreviation		Page
CD	Constant Density	41
RU	Random Uniform	42
LAN	Local Area Network	43
ANOVA	Analysis of Variance	43
IID	Independently and Identically Distributed	44
GER	Global Energy Ratio	54
ln(ATB)	Natural Log of Average Transmitted Bits	84
ln(ARB)	Natural Log of Average Received Bits	84
ATB	Average Transmitted Bits	88
ARB	Average Received Bits	89

Evaluation of Energy Costs and Error Performance of Range-Aware, Anchor-Free Localization Algorithms for Wireless Sensor Networks

I. Introduction

Sensor data without complete coordinates...is next to useless. [SRB01]

1.1 Motivation

As processor, circuitry, and wireless networking technology advances, the interest in (WSN) increases all the more. These advances enable the use of cheap, small, low-powered, computing devices to sense events or phenomena and transmit that information via wireless communication. Collections of these wireless nodes form a WSN and have great potential uses in a wide variety of fields. The ability of these networks to collect data over a vast area, and correlate and aggregate the information simultaneously provides a unique and powerful capability. Wireless sensor networks are being deployed for applications such as target tracking, intrusion detection, environmental monitoring, climate control, and disaster management [MGZN03]. With such data gathering capabilities, wireless sensor networks show great promise for military applications as well. For example, WSNs can gather intelligence over vast areas that are difficult to monitor via conventional means. A thousand wireless sensor nodes each with a 50ft (15.24m) sensing range can be deployed over a 1 square kilometer area to monitor and detect movement. The information can be collected, correlated and relayed in near real time to observers far from the monitored location. This capability enables remote, unobtrusive, continuous monitoring of difficult or hard to access terrain, or tracking border crossings in mountainous or desert environments. However, for wireless sensor networks to be used in military environments, nodes need to be durable, cheap, flexible, low maintenance and self sustaining over long periods of time.

Crucial to any wireless sensor network application is the ability of a node to know its position either globally or relative to other nodes. The process to determine this location information is called localization. To be useful, the localization process must provide accurate position estimates. At the same time, it should be energy efficient to promote long network lifetimes. Ideally, localization does not restrict network flexibility by adding additional hardware or specialized nodes to determine position. Balancing these different needs drives the development of wireless sensor networks, and their uses and is the subject of much of the current wireless sensor network localization research [TP03] [NN01] [SPS02].

1.2 Background

Wireless sensor nodes have a variety of features, benefits and drawbacks. Sensor nodes use different power sources, sensing hardware, and communication capabilities. In general, most have some processing capability, an RF transceiver, and a sensor. Specialized nodes may also have hardware with Global Positioning System (GPS) capability providing global coordinates, making them ideal anchors or beacons for other nodes. GPS is useful for localization, but cannot be used everywhere and is costly. For example, GPS does not work indoors or in areas with significant terrain or foliage. Furthermore, GPS increases the cost, energy usage and platform requirements of the node. Anchors may have their positions set manually, but this method does not scale well. In addition to communication, the RF transceiver can also provide coarse position estimates via a Received Signal Strength Indicator (RSSI). This method converts received power to a distance estimate. A major benefit of this method is it is always available with nodes possessing an RF transceiver. More accurate measurements can be made with acoustic or ultrasonic ranging, but these methods require additional specialized hardware and again increase the cost and energy consumption of the node.

There are six major classes of localization algorithms [PBDT03] [LSS04b]. Range-Aware algorithms use distance or angle information gained from ranging to estimate

position. Range-Free methods do not use this information, but typically use other data such as connectivity and hopcounts to estimate positions. Anchor-Based algorithms use anchors (nodes that have a priori knowledge of their position) in the localization process. Anchor-Free methods do not. Incremental algorithms localize very few nodes at first, then increase the area of localization node by node. Concurrent, or distributed, algorithms localize all nodes simultaneously. Each category has different advantages and disadvantages, and the choice of any one type depends on the application being used, the cost, energy, and hardware constraints, and performance expected from the network.

The performance of any algorithm depends heavily on the operational environment which includes several network and node parameters. For example, the amount of expected error in each distance estimate is critical to the accuracy of range-based algorithms. Network size can influence the performance of incremental algorithms. Since a node's position estimate depends on those nodes which localized previously, a large network may have more errors in nodes at the edge. Nodes near the edge accept the errors of all the nodes localized before them. The node degree is the number of one hop neighbors a node has. Degree is a network parameter that can affect algorithms that depend on having many neighbors. Wireless sensor networks can be deployed in a variety of ways. Grid placement makes localization very easy, but deployment is time consuming and not flexible. Random placement is easy to accomplish, but can result in networks that are not connected or have bad features, such as inconsistent connectivity, which can cause some nodes to have many neighbors and others to have very few. More constant distribution of nodes would be beneficial but would also require significant control over the deployment process.

1.3 Research Focus

The most useful localization algorithms have flexible deployment requirements, limit cost and specialized hardware, and are energy efficient while providing accurate results. The type of algorithm that requires the least amount of specialized hardware

are anchor-free methods relying on a ranging method such as RSSI. This type of algorithm assumes all nodes are identical, and no specialized anchor nodes are needed. Knowing the performance tradeoffs between energy consumption, accuracy, and the major factors influencing those areas would help identify localization algorithms and configurations for deploying rugged, long term, flexible wireless sensor network applications. To accomplish this, an incremental and concurrent range-aware, anchor-free algorithm are evaluated in several environments that vary network size, degree, range error and deployment method. The amount of energy used and accuracy achieved is collected, analyzed and compared.

1.4 Objectives

This research has three major goals. The first is to determine which factor most influences the energy consumption of incremental and concurrent algorithms. Knowing this factor identifies what is most important to control in the deployment of flexible energy efficient wireless sensor networks. The second goal is to determine which type of range-aware, anchor-free algorithm provides the best position accuracy and energy efficiency. The final goal is to build a model that predicts the expected amount of energy consumed when using a particular algorithm under certain network and node conditions. The model allows the energy performance of similar algorithms to be evaluated in conditions not specifically tested in this research.

1.5 Approach

The amount of energy a node uses is the sum of energy expended while sensing, transmitting, receiving, and processing. Sensing is not used during localization and can be ignored. To evaluate energy costs, the number of transmissions and receptions and the amount of processing time each node uses to localize is determined. Since Anchor-Free algorithms do not produce absolute position estimates, the resulting positions could be rotated and translated from the original reference. Instead, the internodal distances are compared to determine the accuracy of the resulting position

estimates. Accurate position estimates will have the same pair-wise distances between all nodes. The impact of various network characteristics on energy consumption is measured by how much each affects the number of transmissions, receptions, and processing an algorithm needs to localize. To accomplish this, algorithms of different types are modelled and simulated under various network conditions.

1.6 Summary

The primary focus of this research is to determine the most important factors influencing energy consumption in wireless sensor network localization. Two particular types of algorithms are examined for accuracy and percent localized. This research identifies the energy consumption characteristics of those algorithms. The Energy Consumption Model predicts energy consumption without simulation or deployment of actual nodes.

The remainder of this document is organized in the following way: Chapter 2 provides an overview of wireless sensor networks, ranging techniques, localization algorithms, their performance and current research in the area. Chapter 3 describes the method used to evaluate each algorithm. Chapter 4 presents and discusses the experimental results and analysis. Chapter 5 summarizes the research, including important results and conclusions and also covers areas of future work.

II. Wireless Sensor Network Localization Background

2.1 Introduction

This chapter presents an overview of wireless sensor network localization algorithms. Range-aware and range-free techniques are discussed as well as their corresponding benefits and drawbacks. Anchor-based methods and issues are briefly covered, while anchor-free methods are discussed in some detail. Incremental and concurrent methods are described and compared. This chapter ends with a comparison of the energy and error performance of current localization techniques and a discussion of current energy reduction efforts.

2.2 Wireless Sensor Network Localization

The recent interest and growth in wireless communication and ubiquitous computing has resulted in a corresponding interest in wireless sensor networks. A wireless sensor network (WSN) is a collection of small, low power, limited computing capacity nodes that sense phenomenon such as temperature, humidity, light, and/or movement and transmit sensor data to neighboring nodes and ultimately the user of the network. These types of networks have been deployed for numerous applications including target tracking, intrusion detection, environmental monitoring, climate control and disaster management [MGZN03]. The ability of a sensor network to sense, collect, correlate, and aggregate data in parallel is one of its inherent strengths.

Wireless sensor networks applications often must know the position of nodes in the network to operate. For example, some routing protocols depend on node position to determine usable routes [LJD+00] [APL99]. Other protocols and sensor query processing systems rely on node geographic information from a common coordinate system [PBDT03]. Furthermore, information gathered from each sensor may only be useful if the sensor location is known. Consider an intrusion detection system implemented using a sensor network with no location information. This system would be able to determine the presence of an intruder but not the location of the intrusion. Determining this position information is known as localization. The objective of

localization is to determine either virtual or physical coordinates for each node in the sensor network. When nodes are placed in known locations, localization is merely recording this data. This option is not available for ad hoc networks, or in networks where deterministic placement of nodes is not possible. The process is complicated by factors such as limited processing capacity, memory, and energy. Furthermore, ranging techniques like ultrasonic ranging, are complex and error prone.

Each localization technique approaches the problem in a different way and can be classified according to its use of radio range information or whether it uses one or more beacons or anchors. Most localization algorithms have two basic phases. The first is distance (or angle) estimation or ranging. In this phase, a node's distance from other nodes or its angle relative to other nodes is determined. The second phase is location estimation. In this phase, distance or angle information from several nodes is combined to produce a location estimate. Techniques to determine position include triangulation, trilateration, iterative multilateration, and collaborative multilateration, or n-hop multilateration.

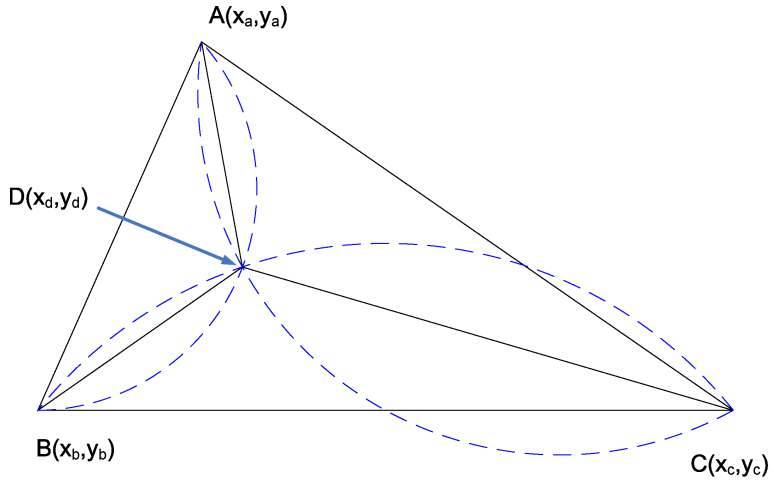


Figure 2.1: Triangulation. Positioning by measuring angles to anchors. [NN03]

Triangulation uses estimated angles between nodes and trigonometric axioms to determine node location. Consider, for example, Figure 2.1. If the angles $\angle BDA$,

$\angle ADC$, and $\angle CDB$, are known, D's position can be found by calculating the intersection of the three circles defined by the anchors and the known angles [NN03].

Unlike triangulation, trilateration uses the measured distance between the subject and each reference point. Trilateration uses these distances and the known locations of the references to determine the location of the node. With distance and location information of at least 3 non-collinear references, a node can uniquely determine its position [SRL02]. In Figure 2.2, m is surrounded by anchors, a_1, a_2, and a_3. When m determines r, its distance to a_1, it can conclude its position is somewhere on the perimeter of the circle centered at a_1, with radius r. When the distance to a_2 is found, m determines its position is one of two points, the points where the circles of a_1 and a_2 intersect. To find the unique location, the distance to the third anchor is needed. The intersection of the three circles defines m's location. If more than three anchors are used, this is atomic multilateration. This can be extend to the three dimensional case by using a fourth reference and intersecting spheres.

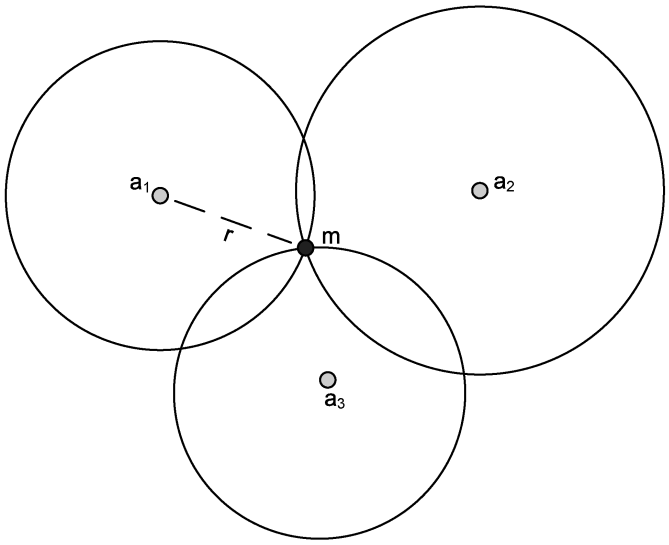


Figure 2.2: Trilateration. Positioning by measuring distances [SRL02]

Iterative multilateration extends trilateration and applies the process iteratively to more and more nodes as their locations are determined. Initially, each node estimates its position with trilateration. Some nodes may not have at least three anchors

in range to estimate their position. Once a node does estimate its position, it acts as an anchor to nodes without a position estimate. Nodes without a position estimate try to localize again with the new anchors. Each iteration yields new anchors that can be used to localize other nodes until the graph is fully localized or all nodes that can be localized with this process have been [SPS02]. Consider Figure 2.3a where node 1 is able to localize itself using the location and distance information from 3 anchors (in grey). Node 2, however, is only in range of 2 anchors and cannot localize. Once node 1 has its position information, it acts as an anchor to node 2, allowing it to localize, as shown in Figure 2.3b. Although, this method is simple and computationally inexpensive, it is subject to errors that propagate from one iteration to the next. It is possible some nodes will not localize in sparse regions of the graph.

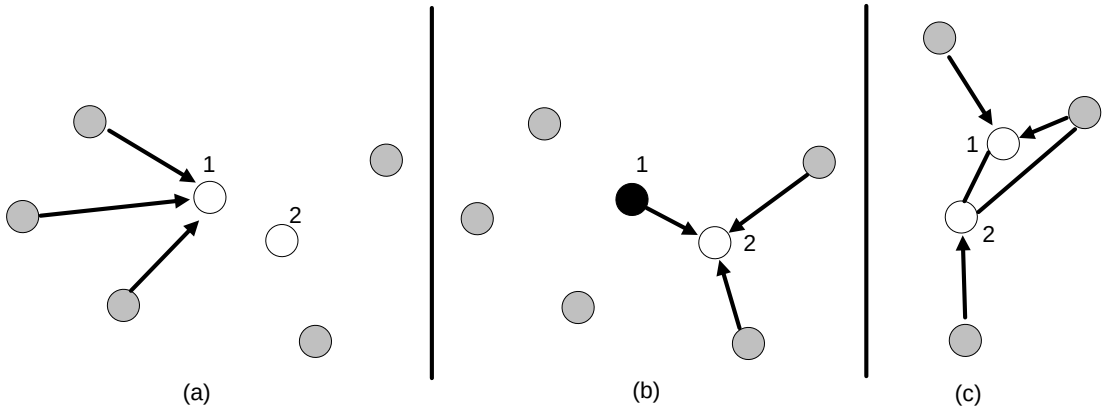


Figure 2.3: Iterative Multilateration (a,b) and Collaborative Multilateration (c) (adapted from [Sav04])

Collaborative multilateration, also known as N-hop multilateration, also localizes nodes without enough anchors in range. This technique needs 3 non-collinear neighbors with unique positions. If a node doesn't have three neighbors with unique solutions, neighbors are recursively called to determine if their positions are unique. Nodes that use the other as a reference must have at least one unique reference node between them [SPS02]. Figure 2.3c satisfies the criteria for collaborative multilateration. Both unknown nodes 1 and 2 have at least one independent reference and three neighbors with unique non-collinear, positions. Once the node determines its location

can be estimated, it bounds the possible area of location in the x and y axis with distances to its reference. The center of the area is chosen as the position estimate as illustrated in Figure 2.4. B's coordinate on the x axis is bounded by the distance from A and the 2 hop distance to C. The same is done in the y axis. Sometimes a third phase refines or optimizes an estimate to further reduce location error.

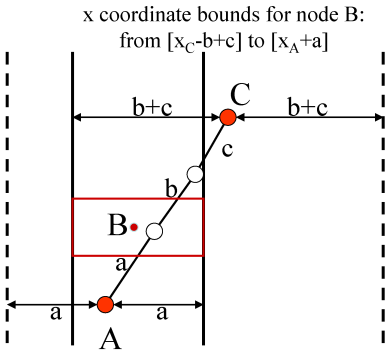


Figure 2.4: Collaborative Multilateration position estimates [SPS02]

2.3 Range-Aware

Range-Aware or range-based localization methods use point to point distance or angle estimates to determine location information. Distance is estimated by measuring, for example, signal strength, signal propagation time, or incident angles. Other algorithms use ranging techniques indirectly to determine the network topology. These algorithms are known as partial range aware algorithms. Range-aware methods are susceptible to ranging errors induced by changes in humidity, temperature and other factors.

2.3.1 Received Signal Strength Indicator. The Received Signal Strength Indicator (RSSI) is a measure of the "strength" of the radio signal at the receiver. Given a known transmission power, effective propagation loss can be calculated at the receiving node. Theoretical and empirical models convert this propagation loss into an estimated distance. This method has the advantage of using hardware that is typically on a wireless sensor node already and therefore does not include expensive

hardware or energy costs. However, RSSI is susceptible to errors caused by background interference, multi-path fading, and irregular signal propagation properties. Despite the potential error, the low cost and availability of the technique make it an attractive option. RSSI is used in several systems such as RADAR [BP00], "Motetrack" project [LW05], SPOT ON [HWB00], LANDMARC [NLLP03] and the Ferret system [TGB+04]. These systems achieve accuracies on the order of one meter using RSSI. The Ad Hoc Positioning System (APS) also uses RSSI as one of its ranging techniques [NN01].

2.3.2 Time of Arrival, Time Difference of Arrival. Time of Arrival (TOA) uses signal propagation time and a known propagation rate to determine distance. Robots use this principle to determine the distance to obstacles by measuring the time of flight of an ultrasonic signal bouncing off of them [Enc97]. Aircraft altimeters use the time it takes an electromagnetic signal to reflect off the ground to determine altitude. Global Positioning System (GPS) uses TOA as well. Similarly, Time Difference of Arrival (TDOA) uses the difference in signal propagation time to several destinations or the different arrival times of different signals to the same destination to estimate distance. Several different types of signals can be used with these techniques, such as RF, acoustic, infrared, and ultrasound. For example, if a node sends an RF signal and ultrasound signal at the same time, the receiving node can infer the range using the difference in the arrival times of the two signals and the known propagation rates of sound and light. RF transceivers are often incorporated into wireless sensors, but other types of receivers usually have to be added to be able to use the difference of two signals as the ranging technique. The additional hardware uses more energy and adds cost and complexity to each node. Unfortunately, ultrasonic techniques also greatly reduce detectable range and increase the required node density. However, when the Ad Hoc Location System (AHLos) was implemented using both RSSI and TDOA (with RF and ultrasound signals), the TDOA version produced more accurate results and was less prone to error due to physical effects [SHS01]. The Cricket indoor

location support system, developed at the Massachusetts Institute of Technology, was originally designed to use RSSI to determine distance to the nearest beacon [PCB00]. However, this produced poor results because RF propagation characteristics within buildings vary too greatly from empirical mathematical models. Instead, the beacons were built to concurrently transmit a RF and an ultrasonic pulse. The time difference between the receipt of the first bit of the RF signal and the ultrasonic signal determines the distance to that beacon.

2.3.3 Angle of Arrival. A technique known as Angle of Arrival (AOA) uses the incident angles of signal reception at multiple nodes to estimate distance. This requires nodes to sense the direction a signal arrived from. Either an antenna array or several ultrasound receivers sense the angle of arrival which also provides orientation information to the node. Antenna arrays would likely be prohibitive in size and power consumption, but nodes with multiple ultrasound receivers have been developed [PCB00]. The angle of arrival is determined using the known distance between each ultrasound receiver, a designated node axis and the distance from the receiver to the signal source [NN03]. A recent version of APS [NN03] uses AOA. However, like TOA and TDOA, AOA is expensive in terms of hardware and energy; therefore it is a less common ranging method.

2.3.4 Acoustic. Acoustic ranging uses the inverse square relationship between the intensity of a transmitted acoustic signal and the distance at which the signal is received. The decay model depends on the size and shape of the source, the surrounding environment and the frequencies of the propagating sound. Other factors that influence the model include background noise, obstructions, wind strength and direction, and foliage [SH03]. An environment with sufficient amounts of any of these factors will degrade the performance of this localization method. Another disadvantage is the additional hardware required to send and receive acoustic signals. This method has been used in various systems for wireless sensor network localization as well as target tracking [SH03] [CYE+03] [LH03].

2.3.5 Partial Range Aware. Since a range technique may introduce significant location error for a given network, ranging techniques are sometimes used to provide information about the network without using the range as a direct estimation for distance. Partial range aware techniques do not use a ranging method for distance estimation but as inputs to an algorithm to estimate locations. For example, the Range Quantization (RangeQ) algorithm [LSS04b] uses partial range information (PRI) to estimate locations. PRI is any type of measurement which monotonically increases or decreases and has an unknown or environment dependent one-to-one relationship with the range measurement. For example, if the actual relationship between RSSI and distance is unknown for a given environment, PRI can be used instead.

Consider a known RF range of a node. Each RSSI value can be mapped to a certain quantization level. Once a node knows the quantization level of its neighbors, it compares and orders its neighbors by distance. This quantization level does not correspond to a particular distance but can be used to order nodes by distance from a particular node. The simplest way to map PRI to quantization levels is by using a linear model. For example, if 2 quantization levels are used and a node has 8 neighbors, the nodes with the 4 highest RSSI values will be have their distance set at 1/2 of the node range. The remaining 4 nodes will have their distance set at the node range. Another method to determine a quantization level constructs a linear model between the maximum and minimum received PRI.

Received PRI values can also be distributed proportional to area. The benefit of this approach is it does not require extra hardware to implement, and the results can be used by other shortest path distance based algorithms [LSS04b].

2.3.6 Range Errors. Each method of acquiring a range estimate is subject to different error conditions. For RSSI, background interference causes signals to be distorted or lost before being received at the destination nodes. In environments with walls or obstacles, a signal may take multiple paths before being received at the destination. If the different signals are in phase, they reinforce each other and

result in a stronger signal. However, if they are out of phase, the received signal will be weaker. A weaker received signal corresponds to a larger distance. Some transmitters may have irregular signal propagation properties and don't fit empirical models built for RSSI. In practice, some empirical studies [GKW+00] [ZG03] [ZHS04] have found that in most environments, RF signals are not isotropic, and there is little or no correlation between signal strength degradation and the distance an RF signal travels. Additionally, when a node runs low on available energy the transmitted power will be less, changing range estimates. These factors result in an accuracy of 2 to 3 meters from a maximum range of 10 meters [Sav04].

TOA and TDOA are less error prone, but require additional hardware and require a shorter inter-nodal range than RSSI networks. This smaller range equates to a much higher required node density, which limits the utility and flexibility of the network. TOA and TDOA are not susceptible to the error conditions described above and enjoy an accuracy of 2 to 5 centimeters at a range of a few meters.

AOA has smaller errors than RSSI but is limited to the range of ultrasound signals, creating an accuracy of about 5 degrees with a range of a few meters [NN03]. Acoustic ranging is susceptible to many errors. Background noise can mask acoustic transmissions. Plants in the environment absorb and interfere with the acoustics causing errors. Furthermore, wind alters the signal by changing its course, or by making it weaker. These factors result in a range of tens of meters, with an accuracy of 10 cm [NN03]. Efforts to mitigate these errors include robust range estimation [GE01], two-phase refinement positioning [SPS02, SRL02], and parameter calibration [WC02].

2.4 Range-Free

Range-Free localization methods make no assumptions about the availability of distance information. This approach is used when coarse accuracy is sufficient because it eliminates the hardware, processing and energy costs many range-aware algorithms require, at the cost of granularity. Some WSN applications use range-free solutions to the localization problem when the costs of the hardware required by range-based

solutions may be inappropriate in relation to the required location precision. Since range-free solutions don't use node distance information, they generally require anchor nodes. Range-Free algorithms determine location information for the remaining nodes in a variety of ways.

2.4.1 Centroid Algorithm. The Centroid algorithm [BHE01] uses anchors to periodically transmit known location information to all neighbors within range. If a node receives enough of these messages, it determines it is in the range of the beacon. After determining which beacons it is in the range of, the node calculates the average of all the x and y coordinates of the beacons in its neighborhood. This is considered the centroid and is used as the location for the node. In this case, the range to the beacon is not estimated or used, but the known transmission range of the beacon is. The advantage of this algorithm is it is simple and easy to implement.

2.4.2 DV-HOP. Distance Vector Hop (DV-Hop) [NN03] is a type of APS [NN01]. DV-Hop uses a classic distance vector exchange so each node learns how many hops it takes to get to each anchor. The node maintains a table of known anchors and the distance to the anchors in hops which it exchanges with its neighbors. When an anchor receives hop distances to other anchors, it estimates the average distance per hop based on its position and the position of the anchor in the received update and disseminates it as a correction to the network. This correction replaces any previous average distance per hop values a node has. For example, in Figure 2.5 the anchor L1 computes a correction of (100+40)/(6+2) = 17.5 while L2 computes (40+75)/(2+5) = 16.42 and L3 computes (75+100)/(6+5)=15.90. Nodes that receive more than one correction, use and forward the first one received and discard the rest. This ensures most nodes receive only one correction, usually from the closest anchor. Using the average distance per hop and the known location of the anchors, the node triangulates its own location. The advantage of this algorithm is its simplicity and it is not susceptible to measurement errors. It is limited to isotropic networks, or networks in which the graph properties (shape and density) are the same in all directions. Some

variants of DV-Hop are DV-Distance, which uses radio signal strength instead of hop count, and DV-Coordinates, where nodes create local coordinate systems and merge them with those of its neighbors. Euclidean propagation [NN03], similar to DV-HOP, uses distance estimates to 3 nodes, at least one of which is an anchor, to calculate location with Pythagora's generalized theorem of triangles. Similarly, Amorphous Positioning uses estimates of hop distance but improves the estimate through neighbor information [Nag99].

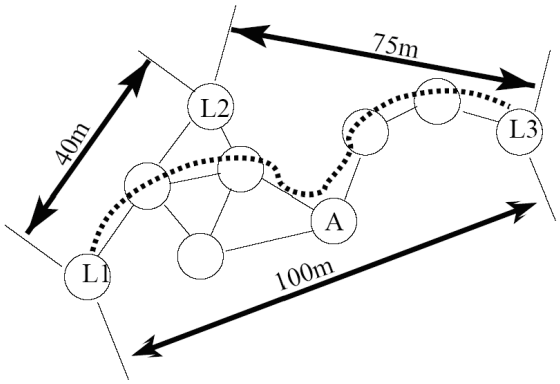


Figure 2.5: DV-HOP correction [NN01]

2.4.3 APIT. The Point in Triangulation (PIT) test [HHB+03] determines if a point is inside or outside a triangle formed by three points with known locations. It does this by "moving" the test point in various directions to see if it moves closer or farther away from one or more of the known points. For example, in Figure 2.6a, any direction M moves it will move closer to either A, B or C. In Figure 2.6b, if M moves in the indicated direction, it will be farther away from A, B, and C. It isn't feasible to implement PIT on a small wireless sensor network, since it would require an exhaustive search of all possible directions and assumes nodes can move.

Approximate Point in Triangulation (APIT) [HHB+03] tests whether a node is inside a triangle of reachable anchors using neighbor information and radio signal strength. If no neighbor of a node is farther away from all three anchor nodes, the test node is likely inside the triangle. For example in Figure 2.7a, none of M's neighbors

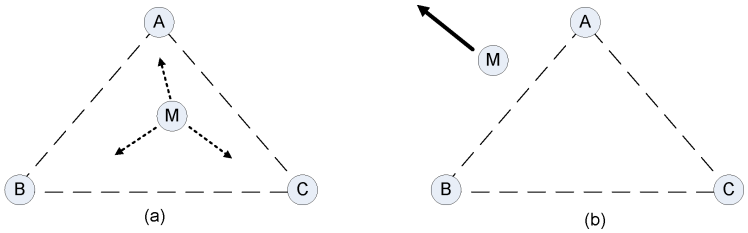


Figure 2.6: Point in Triangulation test [HHB+03]

are farther away from A, B, and C, while in Figure 2.7b, neighbor 4 is farther away from A, B, and C than M.

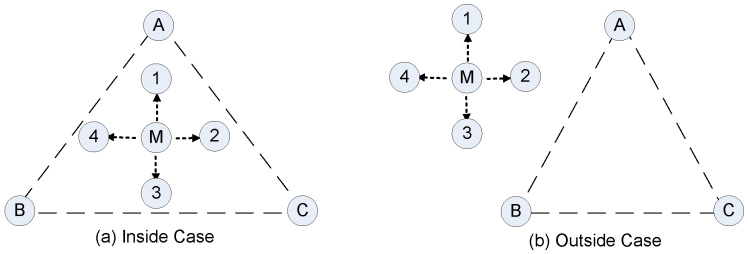


Figure 2.7: Approximate Point in Triangulation Test [HHB+03]

Errors are introduced when a node is inside a triangle but near an edge, and it has a neighbor on the outside of the triangle. A node can mistakenly be considered inside the triangle if all of its neighbors are closer to the three anchors than it is. Once a node has recorded the signal strength and location of all reachable anchors, it exchanges this information with its neighbors and runs APIT on the neighbor data to determine which triangles of anchors it is in. The node aggregates these results by building a map and weighting locations by how many triangles overlap with the location. The center of the area with the greatest weighting is taken to be location estimate. Since APIT does not assume any relationship between signal strength and absolute distance it is considered range-free.

2.5 Anchor-Based

Anchor-Based localization algorithms depend on access to one or more anchor nodes or beacons, with a priori knowledge of their location in a defined coordinate

system. Initial coordinates are typically set manually or using GPS. Anchors transmit location information to the remaining nodes, while minimizing error, communication, and/or energy consumption. The benefit of anchor-based algorithms is resulting node locations are mapped to a global coordinate system. However, these systems are limited in their utility. For some applications, it is not possible to deploy anchor nodes or preplan locations, and GPS can be problematic in some environments. Many anchor-based approaches have already been discussed in this chapter. Other examples include N-Hop Multilateration [SPS02], and Hop-TERRAIN [SRL02].

2.5.1 Global Positioning System (GPS). GPS is a worldwide radio-navigation system formed by a constellation of 24 satellites and their corresponding ground stations. The satellites transmit timing signals with orbit information embedded in the signal. A GPS receiver can determine its location if it is in range of at least four satellites. It uses the received signals to calculate its distance from each satellite, and uses these distances to triangulate its position on earth [Tri05]. GPS, if available, can simplify the localization process and produce a globally-available coordinate system. GPS circuitry is becoming increasingly available on a variety of platforms, including wireless sensors [Cro05]. Even though GPS may be able to fit on some wireless devices, there are still limitations in its use. First, it depends on line of sight communication with the satellites in the network, and so often does not work in indoor and urban environments. Second, GPS hardware and energy costs can exceed what is available or desired on a wireless sensor. If a WSN can be preplanned and manually deployed, but nodes do not have GPS, Walking GPS [SHS04] can be used. This approach uses a commercially-available GPS receiver, a modified wireless sensor, and a simple walk through the network to transmit location information to each node. If a node does not receive a location message, it queries its neighbor and estimates its position as the centroid of its neighbors. Walking GPS produced an average localization error of 0.8 meters. Even though this error is smaller than many other algorithms, this approach would not scale well for large networks and precludes ad-hoc and aerial deployment.

2.5.2 Beacon Placement, Beacon Density. Localization algorithms that use anchors, or beacons, also depend on the placement and density of the beacons. The beacon nodes know their position and serve as a reference for the rest of the network. The density and placement of the beacons controls the granularity of the localization because it determines the size of the localization region [BHET04]. For example, Figure 2.8 shows how increasing the density and placing the beacons closer together, the granularity of the region becomes finer. This results in less error in the location estimate.

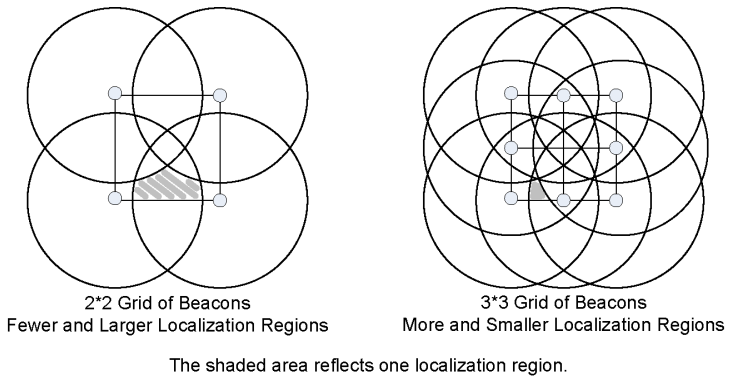


Figure 2.8: Beacon density vs. granularity of localized regions [BHE01]

In many algorithms, the number of visible beacons and their placement is crucial. Furthermore, there may be a minimum number of non-collinear beacons a node needs in order to localize. This leads to several points of failure for the network. A uniform distribution of beacons is an intuitive solution but may be too expensive. The terrain or the environment may preclude this. Finally, if too many beacons are deployed, the risk of collision among signal transmission increases, wasting energy [BHET04]. These issues have led to the development of algorithms that find areas in the network with poor localization and optimal locations for new beacons in an already deployed network [BHE01] [BHET04].

2.6 Anchor-Free

Anchor-Free localization algorithms provide location information without anchors using virtual coordinates. Since these algorithms typically rely on ranging measurements or distance estimates, they generally have a larger position error. If global coordinates are required, they can be obtained from an anchor-free solution if at least 3 nodes know their actual position in global coordinates. Additionally, many applications using wireless sensor networks do not require the accuracy of anchor-based algorithms so much as they require efficiency in power consumption, flexibility, robustness, and simplicity [MGZN03]. Even though there are fewer anchor-free algorithms than anchor-based, anchor-free solutions use a variety of methods. Anchor-Free examples include Self Positioning Algorithm (SPA) which uses local coordinate combining in mobile networks [CHH01], cluster based localization [IS03], fold free forced based relaxation, named anchor-free Localization (AFL) [PBDT03], its refined version anchor-free Localization with Refinement (AFLR) [MGN03], Assumption Based Coordinates (ABC) [SRB01] and Iterative Quality Based Localization (IQL) [EBD+02].

2.7 Incremental

Incremental algorithms start with a small number of nodes that have determined their location or positions relative to each other. They "increment" by repeatedly adding to the set of estimated nodes using the estimated distances to nodes that already have a position estimate [PBDT03]. The new position calculations are achieved by triangulation, trilateration or multilateration. This approach, however, can propagate errors that often grow as nodes are added which produces poor overall position coordinates. Quality can be improved somewhat by applying a refinement algorithm to the estimates if the algorithm can improve on solutions obtained by local minima. An example of an anchor-free incremental algorithm is ABC [SRB01].

ABC begins with a node, $n0$, selecting three nodes in its neighborhood and assigning them coordinates according to their inter-node distances using $n0$ as the origin. Distances are estimated with RSSI. This $n0$ node incrementally calculates the

coordinates of other nodes using the distances to nodes that have already been calculated. Although this algorithm produces a set of coordinates that are topologically correct, simulations show about 60% average position error, for a range error (RE) of 5% [SRB01] due to errors propagated by the incremental approach.

Map Growing is an incremental algorithm developed to perform well in irregular networks. An irregularly shaped network is a fully covered geometric shape, such as a square, rectangle or circle, with some parts cut out. Examples of irregularly shaped networks are O-shaped and C-shaped. Map Growing begins with a well-connected node $n0$ as the origin. Using RSSI, $n0$ defines the coordinates of 2 of its non-collinear neighbors using internode distances, localizing those neighbors. Once localized, those nodes announce their position so unlocalized neighbors can use that information and the RSSI distance to estimate location through trilateration. The map grows as more nodes localize and announce their position for remaining nodes to use. Unknown nodes that only receive 2 announcements use trilateration to estimate two possible locations for their position. Exchanges of neighborhood information with its 2 localized neighbors eliminate one of the possibilities. Any nodes not localized after a certain time can use 3-6 of their closest neighbors to localize themselves. The resulting coordinates can be transformed to an absolute coordinate system if at least three of the nodes are actually anchors.

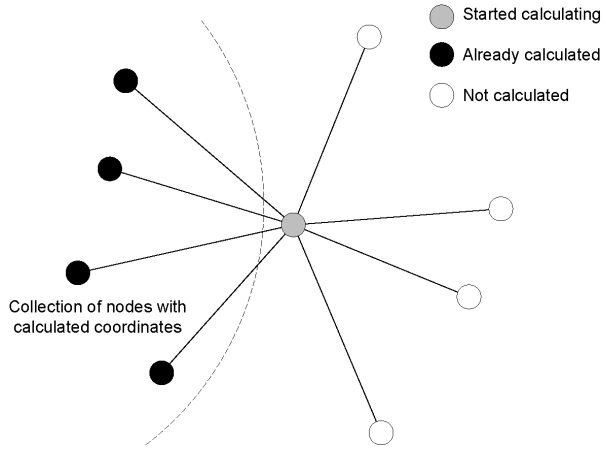


Figure 2.9: Typical incremental approach [PBDT03]

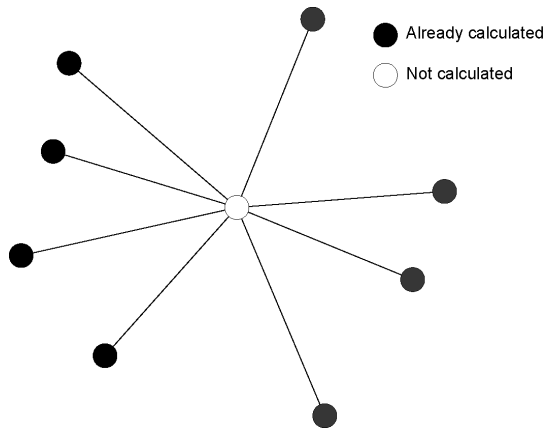


Figure 2.10: Typical concurrent approach [PBDT03]

2.8 Concurrent

In concurrent algorithms, all nodes in the network calculate and refine their coordinate information in parallel. By continually balancing global error, these algorithms avoid error propagation and have a better chance of avoiding local minima, especially in the presence of significant range errors [PBDT03]. In Figure 2.9, an incremental algorithm is compared to a concurrent algorithm in Figure 2.10. In the incremental approach the node distances not yet calculated are susceptible to all the errors created or propagated by the nodes already calculated. In contrast, the concurrent approach calculates all nodes in parallel, so nodes are less susceptible to error propagation.

AFL is a concurrent, anchor-free localization algorithm that uses polar coordinates. The first phase of the algorithm starts by building a "fold-free" graph representation of the network. In a fold free graph, every cycle in the graph has the correct clockwise/counterclockwise orientation of nodes with respect to the original graph [PBDT03]. This type of graph prevents "folds" from inducing local minima solutions that the algorithm's second phase cannot overcome. AFL arbitrarily picks a node and uses hop count to find 4 nodes at the edge of the network which are sufficiently far apart from each other that they form an approximate coordinate system with another node that is approximately at the center. Figure 2.11 is an example of the results of the first phase. Node $n5$ is the origin of this coordinate system and each node calculates its polar coordinates using hop count to $n5$. The algorithm then employs a mass-spring optimization to reduce the errors from the first phase. This optimization compares calculated and RSSI measured distances between a node and its neighbors. The difference puts a "force" between the two nodes, which alters the position estimate in an attempt to balance global "forces". The optimization iterates until the resulting change drops below a threshold. The algorithm has a low probability of converging to a local minima and is able to localize graphs that are at least 6-connected [PBDT03], but at the cost of high communication overhead. The building of the "fold-free" graph has 5 steps where a node must communicate with

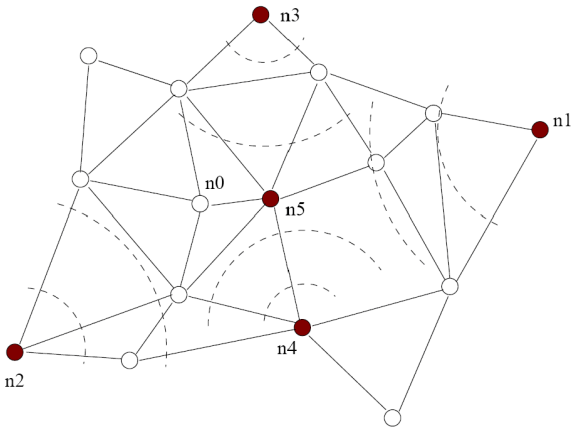


Figure 2.11: Result of AFL Phase 1 [PBDT03]

every other node. Furthermore, the optimization phase may require a large number of neighbor exchanges before completion.

2.9 Related Research

Much of the current research focus is to create or optimize localization algorithms to produce the smallest error. Each algorithm makes separate assumptions and approaches localization in a different way. The localization algorithms discussed thus far are listed in Table 2.1 according to their characteristics. This is by no means an exhaustive listing of algorithms. However, the table is representative of localization algorithms currently being studied or used. In general, range-based techniques are susceptible to various error conditions and will restrict inter-nodal range or require a certain node density. Range-free algorithms typically have the nodes exchange hop count or coordinates instead of measured distances. Anchor-based algorithms are usually more accurate than anchor-free techniques, but at the cost of flexibility. Anchor-free approaches allow networks to be deployed ad hoc, but it is more difficult to find error free position estimates. Furthermore, anchor-free approaches cannot produce node positions mapped to a global coordinate system.

Localization algorithms are naturally judged by the position error they produce, but other factors that can affect the usefulness of the network such as communication

Table 2.1: Characterization of localization algorithms (adapted from [PBDT03])

	Range-Aware		**Range-Free**	
	Incremental	**Concurrent**	**Incremental**	**Concurrent**
Anchor Based	APS(AOA,DV-Dist, D-Eucl) [NN01, NN03], Collaborative Multilateration [SPS02], AhLos [SHS01]	TERRAIN [SRB01]	APS(DV-Hop, DV-Coord) [NN01], Walking GPS [SHS04]	Hop-Terrain [SRL02], GPS-Less [BHE01], RangeQ [LSS04b], Centroid [BHET04], APIT [HHB+03],
Anchor Free	ABC [SRB01], IQL [EBD+02], Map Growing [LSS04a]	SPA [CHH01], AFL [PBDT03], Cluster Based [IS03]		AFLR [MGZN03]

overhead, should also be considered. Of the three power consumption modes, sensing, communication and data processing, communication expends the most energy. It has been shown for short range communication at low radiation power ($\sim$0dbm), energy costs of transmission and reception are nearly the same on a wireless sensor [ASSC02]. If a localization algorithm requires too much communication, the network can have a diminished lifetime after localization is complete. If the communication is not evenly spread, some nodes will fail earlier than others increasing the chance of a partitioned network. When anchors are not available, position error and communication costs become an important trade space. Unfortunately, not all algorithms are tested for communication overhead or energy expended.

In terms of error, ABC by itself does not perform very well. Figure 2.12 compares ABC's performance against TERRAIN (Triangulation via Extended Range and Redundant Association of Intermediate Nodes). Range Error is the percentage the estimated range differs from the true distance. Average Estimated Position Error is the average distance a node's estimated position is from its true position given as a percentage of the range. For a Range Error of just 5%, ABC produces nearly 60% Average Estimated Position Error, while TERRAIN produces about 39%. Even though ABC is simple to implement, it relies heavily on the accuracy of RSSI. Additionally, its incremental approach allows errors to propagate. ABC alone does not refine estimates after they are determined. However, since the output of ABC is a reasonable

starting point, it is sometimes used for initial estimates, as in Iterative Quality Based Localization (IQL) and TERRAIN [EBD+02] [SRB01].

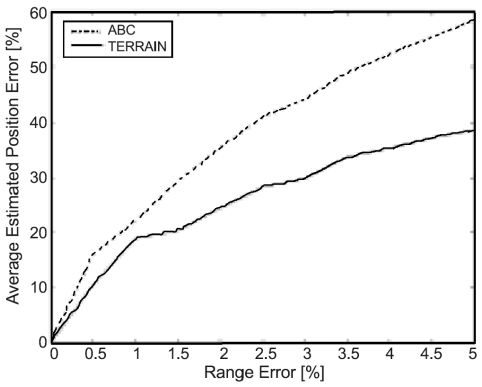


Figure 2.12: ABC versus TERRAIN, 32 nodes total, 4 anchor nodes [SRB01]

IQL uses ABC as an initial estimate for node positions and then runs a Weighted Least Squares algorithm to optimize the positions. Unfortunately, since a node position depends on the accuracy of previously calculated nodes, errors can grow in the presence of inaccurate RSSI measurements, especially at the edge of the network. IQL's error is plotted in Figures 2.13 and 2.14. "S" is a measurement precision constant and reflects the accuracy of the RSSI measurements. When S = 0.2, about 67% of the points have a relative error less than 1. When S = 0.05, 68% of the nodes have a relative error less than 6. Relative error is defined as $\delta x = \frac{\Delta x}{x} = \frac{x_0 - x}{x} = \frac{x_0}{x} - 1$ where x is the true value, x_0 is the computed value, and Δx is the absolute error [Wei05].

The cluster-based approach uses local coordinate systems computed within a cluster of nodes and combines them to form a global coordinate system in a cluster hierarchy [IS03]. The goal of this approach is to reduce the time and communication overhead to localize a network without anchors. Compared to approaches that do not use clusters, the cluster-based approach converges faster and requires dramatically less overhead [IS03]. Figure 2.15 compares the communication overhead of the Self Positioning Algorithm (SPA), which does not use clusters, and the cluster-based approach. In SPA, node convergence includes updates on the angle measurements of all its neighbors to all neighbors. The size of each update is proportional to the number

of nodes in the neighborhood. In the cluster-based approach, each node only sends updates of its master node neighbors. This reduces the size and number of messages required [IS03].

AFL, as a concurrent algorithm, avoids the cascading errors of incremental approaches. When tested and compared against an incremental approach, it localized more nodes with less connectivity [PBDT03]. It also was found to be more robust to measurement errors. The maximum error (the difference between the actual distance and the estimated distance) between two unconnected nodes is a measure of how much the graph has been deformed by the process. In Figure 2.16, the maximum error

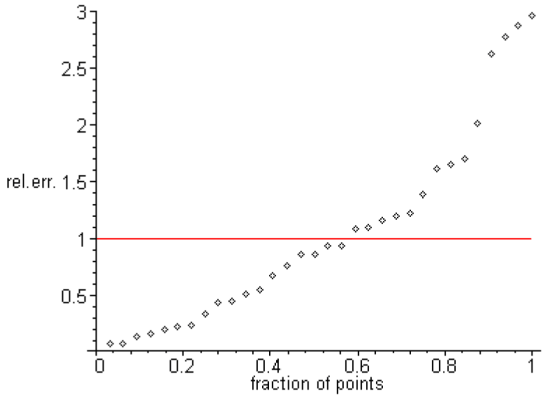


Figure 2.13: IQL Relative error, 40 nodes, S=0.2, 8 anchors [EBD+02]

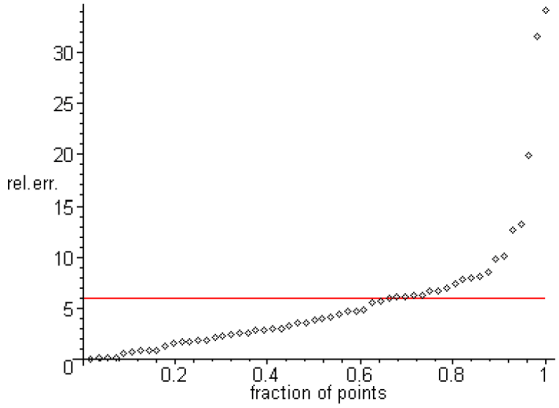
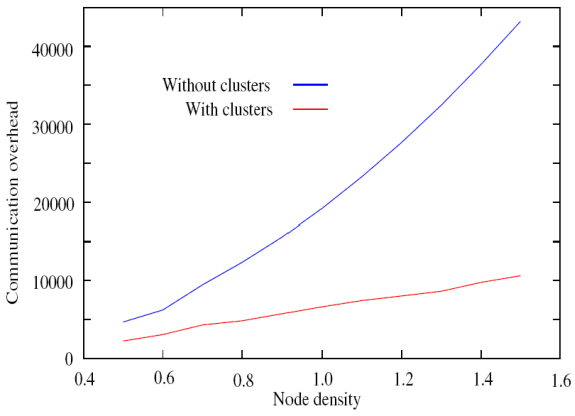


Figure 2.14: IQL Relative error 60 nodes, S=0.05, 4 anchor nodes [EBD+02]



Area: 40 × 40, $r = 2.0$

Figure 2.15: Comm. overhead for cluster-based approach and SPA [IS03]

between two unconnected nodes is compared to connectivity and range measurement error. Since the maximum error is small in most cases, AFL is considered robust even under situations of considerable range measurement error.

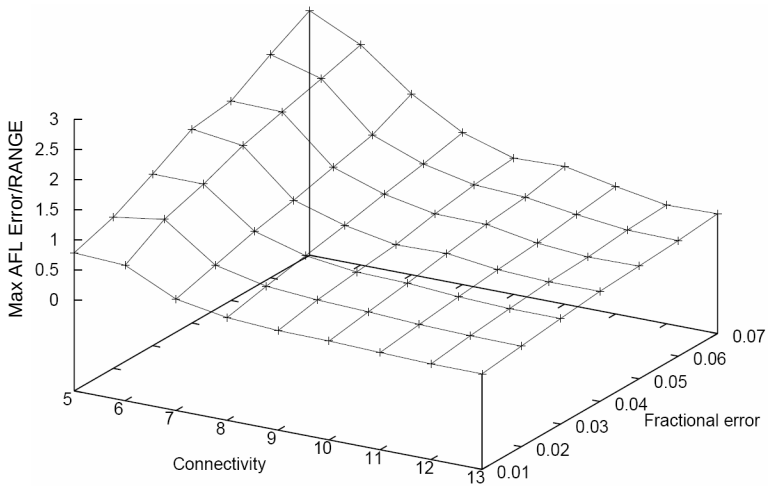


Figure 2.16: AFL: Fraction of $\frac{\text{Max error}}{RANGE}$ between any two unconnected nodes [PBDT03]

Anchor-free Localization with Refinement (AFLR) uses local coordinate systems built by nodes and combines them into a global coordinate system around a "global sink." Since the global coordinate system grows from the center outward, nodes farther from the sink tend to have larger errors than those that are closer. This necessitates a refinement phase where neighbors exchange positions estimates in their local coordinate system. Upon receipt of an update, a node takes a weighted average of the update and others received based on its distance from the sink. After the update, nodes transmit updates to their neighbors. Updates from nodes farther from the sink are discarded. Nodes with 7 neighbors (a degree of 7) have error reduced by 10%. This increases to 30% when the node degree increases to 13. Figure 2.17 plots the improvement versus the average node degree. This localization approach requires each node to have a degree of 9 to localize 90% of the nodes [MGZN03]. Although

position estimates improved after refinement, no measurement of actual position error or relative error or comparison to another algorithm is available for this method.

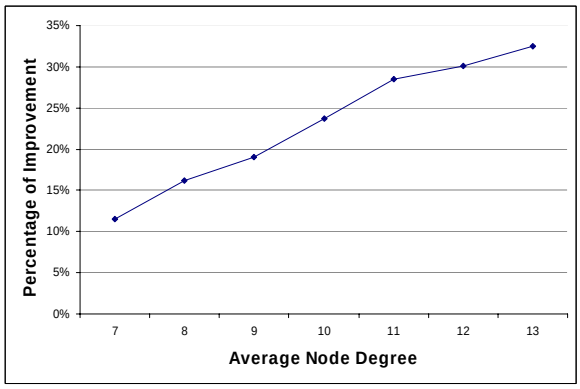


Figure 2.17: AFLR accuracy improvement of refinement [MGZN03]

APIT and APS, both anchor-based algorithms, have been tested with respect to communication costs. The APS study compared the DV-Hop, DV-Distance and Euclidean methods of estimating and propagating positions. The DV-based methods provide good estimates in most cases with the benefit of low signaling complexity [NN01]. The Euclidean-based method is more accurate for nonisotropic networks and is more predictable in performance but at the cost of more communication [NN01]. On average APS produced results within 1 hop from the true position and is usable by geographic routing algorithms. The APIT study compared the performance of Centroid, Amorphous Positioning, DV-Hop and APIT. Because of the flooding DV-Hop and Amorphous Positioning require, the communication needed in those two algorithms grow when either the number of anchors or node density increases. The Centroid and APIT communication needs also grow but at a very much smaller rate [HHB+03].

The performance summary of each algorithm in the study is listed in Table 2.2. Anchors Heard is the average number of anchors heard by a node during estimation. ANR is the Anchor to Node Range Ratio. This is the average distance an anchor signal travels divided by the average distance a regular node signal travels. DOI is

Table 2.2: Performance comparison of 4 anchor-based algorithms [HHB+03]

	Centroid	DV-Hop	Amorp.	APIT
Accuracy	Fair	Good	Good	Good
Node Density	>0	>8	>8	>6
Anchors Heard	>10	>8	>8	>10
ANR	>0	>0	>0	>3
DOI	Good	Good	Fair	Good
GPS Error	Good	Good	Fair	Good
Overhead	Smallest	Largest	Large	Small

the degree of irregularity which is an indicator of radio pattern irregularity. Overhead is the amount of communication used by the algorithm.

The communication cost of the distributed version is compared to a centralized version of the algorithm. This comparison is shown in Figure 2.18. The figure shows that the even though the overall cost is about the same, the communication is more evenly spread in the distributed version. The centralized version has several nodes with very high communication costs and some that use very little communication. This unevenness in communication distribution can cause some nodes to expend their energy sooner than others, leading to some nodes failing earlier than the rest which increases the likelihood of a partitioned network. Since the last part of the process includes small refinement of the position estimate, the higher level application can terminate the position refinement to conserve energy while accepting slightly worse results [SPS02].

2.10 Summary

This chapter introduces and discusses wireless sensor networks and localization concepts relevant to this research. Many approaches exist for localization each with their strengths and weaknesses. The tradeoff space includes error rate, hardware, cost, flexibility, scalability, and energy consumption. Although position error is often most important, significant work has also been done to limit communication in an effort to conserve node energy consumption.

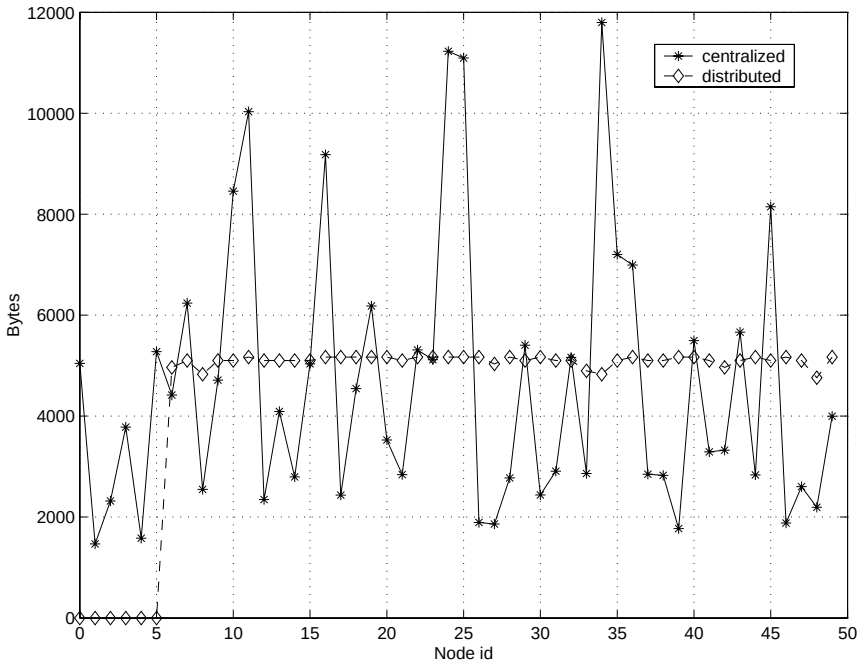


Figure 2.18: Communication cost of Collaborative Multilateration Distributed and Centralized, 6 beacons 44 unknowns [SPS02]

III. Methodology

3.1 Introduction

This chapter discusses the methodology to evaluate the error and energy performance of anchorless wireless sensor network localization algorithms. First, the problem is discussed and defined. Second, the goals and objectives are presented. Next, the system, its services, the workload and metrics are covered, followed by a discussion of the evaluation technique and experimental design. Finally, the technique to analyze the data is covered in detail.

3.2 Problem Definition

3.2.1 Goals and Hypothesis. Wireless sensor networks need node position information to operate and depend on localization to provide it. The most flexible networks can be deployed ad-hoc and without anchors. However, anchorless networks are more difficult to localize and as such use more energy to localize. Energy used during localization is not available for the operation of the network. Additionally, if some nodes in the network are more heavily used during localization they will have disproportionately less energy available after localization. This makes network partitions more likely. The goal of this research is to evaluate the communication and energy costs of anchorless localization algorithms. Specifically, the goal is to determine the conditions that most affect communication and energy cost in localization. The research determines which type of anchorless range-aware algorithm provides the best node position accuracy and the most efficient energy usage. This should identify which kind of algorithms effectively prevent the partitioning of a network, while extending network lifetime. The data from the research also enables a model that predicts energy usage based on several network and algorithm factors.

The different characteristics of localization algorithms and network configurations determine their accuracy and energy performance. In particular, whether an algorithm is Range-Free or Range-Aware, Anchor-Based or Anchor-Free, incremental or concurrent, drives the energy consumption characteristics of the algorithm. Node

degree, the number of nodes a single node is in range of, is also an important factor. The number of nodes in the network will likely impact energy cost and accuracy performance.

3.2.2 Approach. The amount of energy a node expends depends on the amount of sensing, transmitting, receiving, and processing it does.

$$E_{Total} = E_{Transmissions} + E_{Receives} + E_{Sensing} + E_{Processing} \tag{3.1}$$

To evaluate the energy costs of different localization algorithms, the number of transmissions and receptions and the amount of processing time each node uses to localize is determined. Since localizing does not involve sensing, this method of energy consumption is ignored. Since anchor-free algorithms can not produce position estimates in an absolute coordinate system, the straightforward approach of measuring position error cannot be used. The resulting position estimates will likely not have the same origin and orientation as the actual network, but the distances between the nodes can be compared. If an anchor-free localization algorithm is accurate, pair-wise distances between all nodes should be the same. Determining the impact of various network characteristics on energy cost and network lifetime is determined by how much these characteristics change the number of transmissions, receptions, and processing an algorithm needs to localize and the internodal distance error produced by the algorithm. Thus, the algorithm is modelled and simulated under different network configurations and conditions.

3.3 System Boundaries

The System Under Test (SUT) includes the wireless sensor nodes and anchors within a wireless sensor network. These nodes receive, transmit, and process data. The range of the node transceiver affects who it can communicate with and is also part of the system. As shown in Figure 3.1, the transmitter and receiver are assumed to have the same range in all directions, and the reception outside the node range

is nil. Therefore, a node either receives a message completely or does not receive it at all. Depending on the algorithm being studied, a node can estimate its distance away from another node using Radio Signal Strength Indicator (RSSI), the incident angles of reception, or the time of arrival. These capabilities are part of the SUT. Since localization does not depend on a node's sensing capabilities, this is not part of the SUT. Anchors are the same as other nodes except they have a priori knowledge of their location. None of the networks used in the experiment use anchors. The SUT does not include the physical network layer - error free communication is assumed. However, the medium access control (MAC) protocol is included since it controls when and how often a node retransmits a packet and thus affects the energy a node uses to localize. The SUT does not include any application that uses the location estimates. Additionally, all nodes are stationary and are in a two dimensional space. The component under test (CUT) is the localization algorithm the nodes use to find their estimated location. For a given system, all nodes use the same localization algorithm.

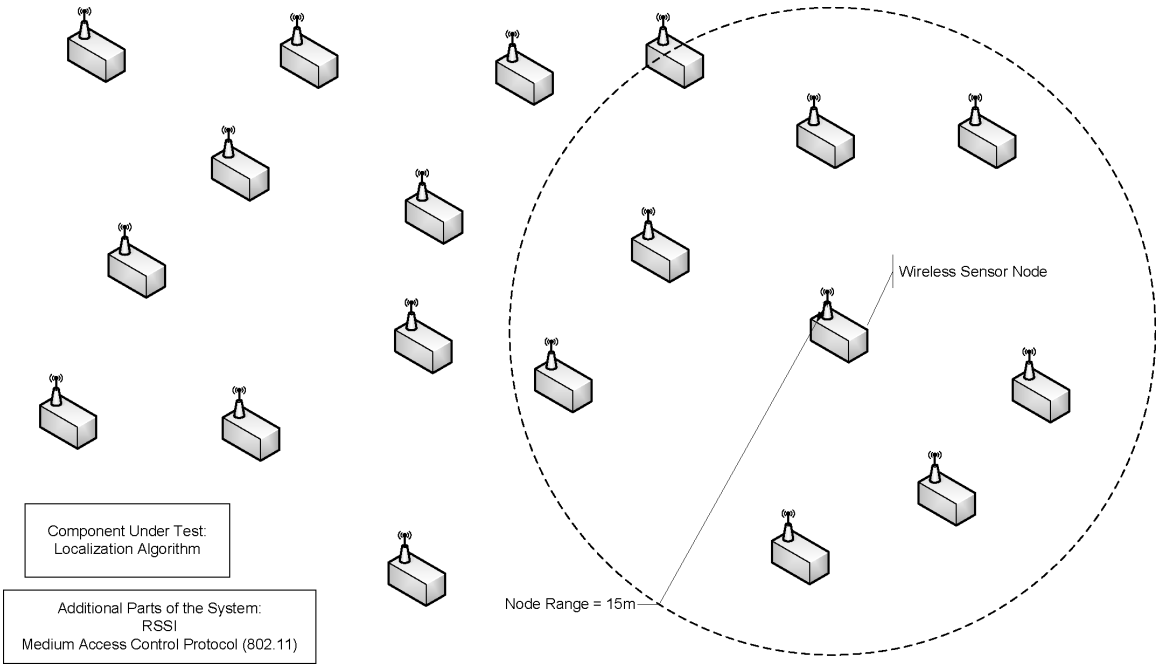


Figure 3.1: System Under Test

3.4 System Services

The service the system provides is a node position estimation service. The ideal localization algorithm localizes all nodes in the network to their position in the shortest amount of time using the fewest messages. At the most basic level, success is defined as a node being able to estimate its position. Failure is defined as a node being unable to estimate a position for itself. Various conditions can cause a failure. Each algorithm has different information it needs for a node to be able to localize. Many require a node to have a certain number of available references (anchors or other localized nodes) before it can be localized. If a node is isolated or does not have enough qualified neighbors due to network configuration or insufficient transmitter/receiver strength, the node will not localize. Not meeting the algorithm's criteria to localize is one cause for failure. If the network becomes partitioned for whatever reason, one or more nodes may not localize. Other failures can be caused by phenomenon that cause wireless traffic to fail, such as network congestion and interference. Network congestion can cause a failure if node transmissions collide with its neighbors often enough to prevent it from communicating effectively. Interference occurs when wireless devices that are not part of the network operate within the same radio frequency. This can also hinder communication enough to degrade or prevent localization. For the purposes of this research, interference and network congestion is not modelled, but all other failures are.

Localization algorithms achieve various levels of accuracy and use various amounts of energy to accomplish the task. The required accuracy of a given network depends on the application using the network. For example, a wireless sensor network built to monitor environmental conditions on a sequoia tree may require more accurate and precise position information than an intrusion detection network. Both need position information, but the intrusion detection network can be useful if position information is accurate to within a couple feet, whereas the sequoia monitoring system would produce unreliable data if off by that much. Alternately, a network meant to operate for a few weeks has much different energy requirements than one meant to operate

for a few months or longer. Even though an algorithm may be successful by the definition proposed above, it may not be sufficient for some applications. Further, given two successful algorithms, one may be preferable for a given purpose depending on the error produced or energy expended. Complete success of a localization algorithm depends on the domain or purpose of the network being built. This research uses the definition of success above, but evaluates and compares how successful an algorithm was based on performance metrics.

3.5 Workload

The workload for the system is the network configuration the localization algorithm must operate within. The network configuration is the set of characteristics that define the network and includes the number and placement of the nodes. The placement of the nodes determines node density, or how many nodes there are per unit area. The placement can be uniform or can be according to a certain distribution, such as a normal or exponential distribution. Another characteristic is the average node degree or the average number of nodes a given node can communicate with. Network shape affects how messages propagates through the network. An isotropic shape is the same or similar in all directions from the center of the network. An example of a non-isotropic network is one in the shape of the letter "C".

These characteristics are considered workload because they directly affect the performance of localization algorithms. The more nodes there are in the network, the more messages a localization algorithm will need to send. In a dense network, more nodes receive each message. Conversely, some algorithms may require fewer messages if more nodes hear each one. If node degree is low, it is more likely some nodes will not localize at all. The placement distribution and network shape affect the required time and energy to localize in a similar way.

3.6 Performance Metrics

The following metrics are used to evaluate the performance of a localization algorithm for a given network.

- Average Distance Error (ADE) - At the end of an anchor-free localization algorithm, each localized node in the network will have their true absolute coordinates and their relative coordinates estimated by the algorithm. Since the relative coordinates will not be in the same coordinate system as the absolute coordinates, it is not possible to use position error as a metric. Position error measures the distance between the true absolute coordinates and the absolute coordinates estimated by the localization algorithm. Instead Average Distance Error is used. Despite the difference in the coordinate systems the distance between any pair of nodes should be the same. The average distance error is the average of all the differences in the distances between a pair of nodes absolute coordinates and relative coordinates. For example, given nodes i and j. The absolute coordinates are (x_i, y_i) and (x_j, y_j). The relative coordinates are (x'_i, y'_i) and (x'_j, y'_j). Between i and j, the distance error e_{ij} is the difference between the true distance (d_{ij}) between i and j and the distance in the algorithm's result $(\hat{d_{ij}})$. Thus, the distance error is

$$\begin{aligned} e_{ij} &= |d_{ij} - \hat{d_{ij}}| && (3.2) \\ &= \left| \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} - \sqrt{(x'_i - x'_j)^2 + (y'_i - y'_j)^2} \right|. && (3.3) \end{aligned}$$

The average distance error is the average of all distance errors between all pairs of nodes, or

$$\text{ADE} = \frac{\sum_{i,j:i<j} e_{ij}}{N} \quad (3.4)$$

where N is the number of localized nodes. ADE can be interpreted as the average distance a node's estimated position is from its true position relative to another node. ADE also has the benefit of being expressed in meters.

- Average Transmitted Bits (ATB), Average Bits Received (ARB) - Energy used in node operations are attributed to transmitting, receiving, sensing and processing. Since the major energy cost in localization is communication, this is collected as part of the energy consumed to localize. The total number of bits sent and received is collected to find the total for the network and divided by the number of nodes to find the average.
- Percentage Localized - Percentage Localized is the percentage of nodes able to estimate a position before the algorithm finished. If a node does not localize, it ca not participate in the operation of the network. The fewer nodes that are localized, the fewer nodes that are available for operation. This is a global network metric.

3.7 Parameters

The parameters discussed below affect localization performance.

3.7.1 System.

- Node Range - The node range is how far a node can transmit and receive packets. This determines the nodes that are in range. If this range is large, a node is able to communicate with more nodes, but the chance of collision is higher.
- Ranging Accuracy - The ranging accuracy is how close a distance, angle, or time difference estimate is to the true value. The higher the accuracy, the less error there is likely to be in the final position estimate.
- Antenna/Link Type - The nodes in this system have omnidirectional antennas with bi-directional links. This means the antenna transmits in all directions, and a node that can transmit to another node can also receive from it. Directional

antennas and one-way links make localization much more complicated and are typically not considered in localization research.

- Algorithm - The particular algorithm and its type (Range-Aware, Range-Free, Anchorless, Anchor-Based) used in a localization system determines how much time, how much error, and how much energy is expended to localize the network.

3.7.2 Workload.

- Number of Nodes - The number of nodes in a network affects how much energy a localization algorithm needs to estimate node positions. For some algorithms, it may also affect position error. This parameter also indicates how well a localization algorithm scales.
- Number of Anchors - For anchor-based algorithms, the number of anchors may affect the accuracy and speed of the algorithm. The more anchors there are, the more likely a node is of being in range of one. With more anchors in range, a node is more likely to have more accurate results, since the anchors have precise knowledge of their position. This parameter can also be defined as a percentage of the total number of nodes in the network.
- Anchors Heard - For anchor-based algorithms, the more anchors heard by a node means that a node is more likely to have enough anchors to localize and estimates a more accurate position. This is directly affected by the number, range and density of the anchors.
- Network Area/Node Density - The area the network covers combined with the number of nodes in the network determine the node density (assuming a uniform distribution). The node density also affects how many neighbors a node has which affects how well a localization algorithm works.
- Node Degree - Node density and node range define the node degree, the number of neighbors a node can communicate with. The higher the node degree the fewer messages required to reach each node, but collisions become more likely. Node degree also affects how many nodes are localized. The lower the node degree the less likely a node meets the criteria to localize.
- Network Shape - The network shape locally affects the node degree in the network. For example, an isotropic network often has a more consistent node degree than one with an irregular shape. An irregularly shaped network likely has more

nodes with lower node degree, causing them to have few neighbors and possibly be isolated.

- Placement Distribution - The distribution of the node placement also affects local node density and node degree in the network. If the placement distribution is uniform, the node degree is expected to be consistent. However, if a different distribution is used some portions of the network may suffer from having too few neighbors or anchors in range. In terms of anchors, the placement affects how many nodes are in range of the anchor and if that placement is beneficial for localization. A node that is in range of 3 anchors that are collinear or close to it, still cannot localize effectively.

3.8 Factors

The following are the different factors used in this research and their corresponding levels.

- Algorithm
  - Map Growing - Map Growing Localization [LSS04a] is a range-aware, incremental, anchor-free algorithm designed to perform well in irregular shaped networks. This algorithm starts with a non-collinear set of three nodes, arbitrarily defines their coordinates, and incrementally adds nodes using range information, and trilateration. Any nodes not localized after this process use 3-6 of their closest neighbors to localize themselves through trilateration. This algorithm was chosen as representative of incremental, range-aware, anchor-free algorithms.
  - Anchor-Free Localization (AFL) - AFL [PBDT03] is a range-aware, concurrent, anchor-free algorithm that tries to avoid false local minimum solutions by creating relative polar coordinates for each node through hop counts and using a mass-spring optimization to refine the estimates. This algorithm represents concurrent, range-aware, anchor-free algorithms.

- Number of Nodes These are typical sizes for small, medium and large networks in WSN research.
  - Small - 30 nodes represents a small network.
  - Medium - 100 nodes represents a medium sized network.
  - Large - 300 nodes is considered a large network.
- Average Node Degree A node degree of 8 is used as a typical value in WSN research. This is taken as low, while other levels are based on that.
  - Low - An average node degree of 8 represents a low node degree.
  - Medium - An average node degree of 12 represents a medium node degree.
  - High - An average node degree of 16 represents a very node degree.
- Range Error Ranging accuracy is often measured as Range Error, or the percentage a distance is from the true distance. For example, if the true distance between 2 nodes is 10m but the ranging method reports a distance of 11m, the Range Error is $\frac{11-10}{10} = 0.1 = 10\%$. The Range Error in the experiment is modelled as a normal Gaussian distribution [EBD+02] with the mean set at 0 error and the level used as the standard deviation. For instance, a Range Error level of 10% results in a normal Gaussian distribution, with the mean at 0, and a 0.1 standard deviation. In this case, 66.7% of the range estimates will be between -10% and +10% of the true distance. Typical experimental values for range error vary from 0 to 10% and is reflected in the levels below.
  - Low - A Range Error of 2% represents low ranging error.
  - Medium - A Range Error of 5% represents medium ranging error.
  - High - A Range Error of 10% represents high ranging error.
- Placement Distribution
  - Constant Density (CD) - The constant density distribution represents networks where deployment is controlled. Using this distribution, the network

area is partitioned into grids and nodes are evenly divided amongst these grids (uniformly random distribution within each grid). The resulting network is also guaranteed to be connected, meaning that every node has at least one communication path to every other node. Appendix A has examples of CD networks for each size and degree.

- Random Uniform (RU) - Random distribution represents networks where there is little control over deployment. Nodes are placed in any possible position in the network using a uniformly random distribution. These networks are also guaranteed to be connected. The major difference is that some areas of the network have a lower density than other areas. Also, there are no guarantees on the minimum or maximum degree of individual nodes. Appendix A has examples of RU networks for each size and degree.

3.9 Evaluation Technique

To evaluate the system, OPNET 10.5A, a network simulation environment, models and simulates the network. Simulation is used for various reasons. Since no general analytical models exist for localization algorithms, one would have to be created to evaluate this system. In addition, the time and resources necessary to create and run actual wireless sensor networks of 30-300 nodes make direct measurement infeasible. The controlled, repeatable environment of a simulation make it the preferred choice for this research.

The algorithms used in the system are designed and programmed in OPNET. The implementation of these algorithms is validated against a manual execution of each algorithm on a smaller network. Furthermore, the results of the experiment are compared to results using similar configurations found in other work [LSS04a] [PBDT03]. Results are not expected to agree exactly, but should have the same trends and general behavior.

3.10 Experimental Design

To evaluate the interaction between all the factors, a full factorial design is used. There are five factors, with 2, 3, 3, 3, and 2 levels. A full factorial design requires $2 \times 3 \times 4 \times 3 \times 2 = 108$ experiments. Sufficient statistical basis for analysis is expected to be achieved with no more than 30 replications. This results in a total of 3240 experiments. Each experiment only needs to be run until the localization algorithm finishes. The network area is determined by the required node degree, while the node range is 15 meters. None of the networks used are irregularly shaped. The nodes are modelled using omnidirectional antennas and bi-directional links. The MAC layer protocol is the IEEE 802.11 wireless local area network (LAN) protocol. Even though 802.11 is used at the MAC protocol, collision, congestion, and retries are not modelled. All messages are received.

The random seed is changed before each simulation run to ensure each is independent. Errors are assumed to be normally distributed. Similar to [HHB+03], a 90% confidence interval is used. Given all the factors of the network that are controlled, the randomness in the node position and degree should have little impact. Thirty repetitions should be sufficient to ensure a small variance.

3.11 Analysis

The analysis of the data supports the goals of the research. To allocate the variation in the energy and error, an Analysis of Variance (ANOVA) on each communication response is performed. This shows if the variance in performance is due to experimental error or real differences in the factors. A computation of effects calculation shows how much each level of each factor affects energy and error. To determine if two levels of a factor are significantly different, a confidence interval for contrasts is used. Finally, a linear regression to predict the energy used given the type of algorithm, network size, average degree, network type and range error will be developed with calculations for confidence intervals on the regression.

To perform an ANOVA and a linear regression, several assumptions are made and can be verified with visual tests on the data [Jai91]. To verify that errors are independently and identically distributed (IID) the residuals are plotted versus the predicted response. The plot is examined for any trends that suggests the errors are not independent. By ensuring no trends in the spread of the residuals, the plot also confirms that the standard deviation is constant. Experimental errors are verified to be normally distributed by examining a plot of the quantiles of the residuals versus a quantile of the normal distribution. A linear plot indicates that experimental errors are normally distributed.

3.12 Summary

Wireless Sensor Networks is an exploding field of study and commerce. Their dependence on location information, however, makes an effective and efficient localization algorithm crucial. The most flexible WSNs are ad-hoc without anchors. Small position error is important but is pointless unless node energy has been properly conserved. This trade space is where algorithms need to be evaluated. The goal of this research is to determine the factors that most influence energy usage and error in anchorless localization algorithms as well as develop a model to predict localization energy consumption. Below is a list of expected results of this research.

- Map Growing, the incremental algorithm, is expected to use less energy than AFL, the concurrent algorithm. Once a node localizes in Map Growing it does not need to communicate unless it is needed by a neighbor in the final phase. This is the benefit of the incremental approach, once a node localizes, it is essentially done (unless a refinement phase is added). However, with AFL, a node may trade position broadcasts with its neighbors many times, depending on the accuracy of the initial estimate and range error conditions. This may lead to high communication costs. On the other hand, AFL's optimization phase makes it a good candidate to achieve less error.

- Low degree networks are expected to need more energy to localize than high degree networks. Since each node is in range of fewer other nodes, it takes more messages to reach all the nodes in the network. With a higher node degree, a single message reaches more nodes, reducing network traffic. Also, high degree networks should be more accurate since each node should have more information to use for localization.
- In general, incremental algorithms are expected to have lower accuracy. Error tends to accumulate in incremental algorithms, causing large position errors at the edges of the network. Concurrent algorithms avoid this problem.
- The larger networks should produce less accurate positions with Map Growing, since the error will be able to propagate more in a larger network. This should not affect AFL, a concurrent algorithm, since error only depends on a node's immediate neighborhood.
- A higher range error should force the accuracy to decrease and may require more communication. A discrepancy in a distance may force nodes to delay localization requiring more messages.
- Map Growing claims to localize 100% of the nodes. AFL only requires connectivity to be localized, so it should have a high localized percentage. Higher range error and lower degrees should negatively impact the percent localized.

This chapter defines the methodology used in this research to evaluate the energy costs and error of anchorless localization algorithms. The node position estimation system, its services, and workload are described in detail. The performance metrics, system parameters, and experimental factors are defined. A complete description of the experimental design and method of analysis is also provided.

IV. Experiments, Data and Analysis

4.1 Introduction

This chapter discusses the energy and position error results from wireless sensor network localization experiments using the Map Growing and AFL localization algorithms. First, the validation and verification of the localization algorithms is discussed followed by an overview of the data collection methods. Next, the error performance of each algorithm is examined with respect to each factor, and the same is done with localization error. The percent localized results are also discussed. Finally, the energy consumption model is presented which provides insight into the key energy consumption finding.

4.2 Validation of Localization Assumptions

To simplify the localization simulations and analysis, several assumptions are made. The transmission and reception range are assumed to be equal and consistent across all nodes. Furthermore, this range is assumed to be equidistant in all directions. Thus, if node A can hear node B then B can hear A. Some research suggests actual node ranges vary by direction and by node [AV04], making it possible for B not to hear A. However, most current WSN localization makes similar assumptions to make simulations and analysis tenable. In actual practice, typical node operation mirrors these assumptions [AV04]. Nodes are assumed to be on a planar region, that is in only two dimensions. While it is possible to deploy a WSN in three dimensions (on hills and buildings as well as the ground), two dimensional deployment is a common case for applications such as environment monitoring and target tracking. Furthermore, at this time, there are only a few localization algorithms that have been extended to three dimensions [SRZF03]. The algorithms used herein only operate in two dimensions. An additional simplifying assumption is that collisions and congestion is not modelled. In the OPNET implementation of both algorithms, IEEE 802.11 is used as the MAC protocol. IEEE 802.11 would not typically be used by wireless sensor nodes since it requires the transceiver to be continuously on. Wireless sensor nodes would more

likely use scheduled based MAC protocols that try to conserve battery power. IEEE 802.11 is used in the simulation instead because OPNET wireless sensor node MAC protocols are not currently available and the power consumed due to localization, not the MAC protocol, is the area of interest. With 802.11, if a message is not acknowledged, the node retransmits until it is acknowledged or the retransmit limit is reached. Both algorithm implementations only count the bits transmitted once, not once for every retransmission. Also, when a node needs to transmit during a time other nodes may transmit, the localization algorithm chooses a random time between zero and five seconds to reduce the number of actual collisions in the simulation. This has the effect of making collisions very unlikely. For example, during the initial flooding phase of AFL to find the five reference nodes, the network averages about 5.17 retransmissions per node (300 nodes, average degree 8, Random Uniform network, 10% Range Error). After, adding the random delay, this is reduced to about 1.27. While collisions occurs in WSNs, trial runs and analysis of the algorithms used has determined it is not a significant factor with AFL and Map Growing. The most common and largest packet sizes and typical transmission rates during localization, along with trial data are also examined and while collisions can occur they do not significantly affect the operation of either algorithm.

4.3 Localization Algorithm Validation

To compare the performance of the AFL and Map Growing localization algorithms, each algorithm is implemented in OPNET 10.5A. OPNET is a network simulation environment which allows a user to set up experiments from predefined models and alter various model and network parameters. Although OPNET has several models for ad hoc wireless networking, none are currently available for wireless sensor networks, let alone those that implement localization algorithms. Thus new OPNET models are developed, implemented, and validated against the published results of each algorithm. OPNET is used for various reasons. It is a widely used environment, allows great flexibility in development and statistic collection, and has

built in functionality to handle node deployment and packet transmission. In implementing each localization algorithm, the goal is to realize an algorithm whose behavior exhibited the same trends as the target algorithm. For example, when implementing Map Growing, the goal is to produce an algorithm that is range-aware and incremental and whose error and performance follows the same trends as the published results. Statistically identical behavior would be ideal, but detailed descriptions of the algorithms have not been published so certain assumptions had to be made.

4.3.1 Map Growing. The published Map Growing results [LSS04a] use several different types of networks and range error conditions to illustrate performance. The two used to validate the OPNET implementation (referred to as OPNET-MG when compared to the published results) are 1) 100 nodes placed in a grid with some variance in the position resulting in an average degree of 6.47 and 2) 200 nodes placed randomly in a 100m $\times$ 100m square resulting in an average degree of 12.35. The published Map Growing results from both scenarios are compared against OPNET-MG in Figure 4.1 and 4.2. In both scenarios, OPNET-MG was run on 30 randomly created networks that were created using 30 different random seeds. Figures 4.1 and 4.2 are plotted with the 95% confidence interval of those 30 repetitions. In Scenario

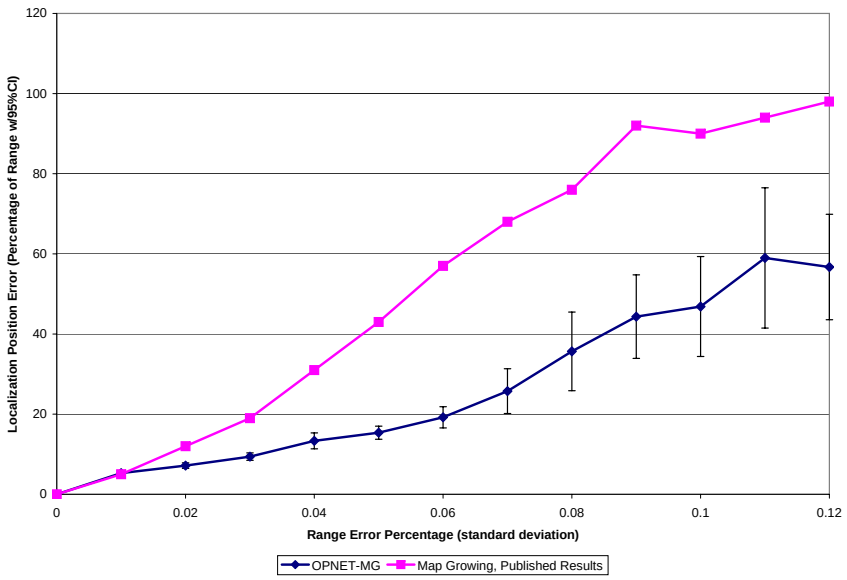


Figure 4.1: OPNET-MG Verification, 100 Nodes, Grid with variance

1, the variance in the grid positions is implemented by creating networks with nodes positioned at multiples of 10m apart in both the x and y axes and adding a value to each coordinate drawn from a normal distribution with $\mu = 0$ and $\sigma^2 = 0.25$. This results in a network with an average degree between 6.36 and 6.54, and the average degree of all 30 repetitions is 6.45. The networks for Scenario 2 are generated in the same manner as described in [LSS04a]. The resulting networks have an average degree between 11.74 and 13.18, and the average degree of all 30 repetitions is 12.655.

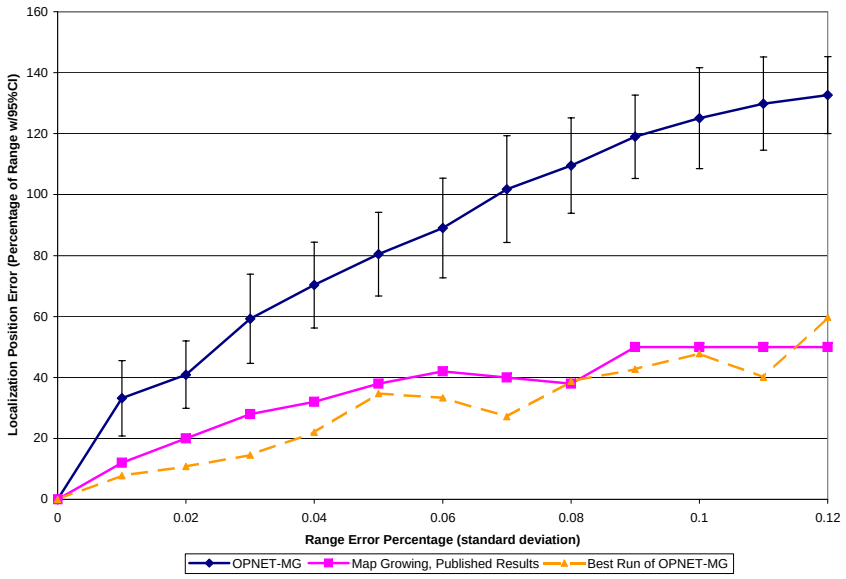


Figure 4.2: OPNET-MG Verification, 200 Nodes, Random Placement

In Scenario 1, OPNET-MG consistently results in a lower position error at range error levels of 0.02 and higher and is identical below 0.02 range error. However, in Scenario 2, OPNET-MG consistently results in a higher position at range error levels of 0.01 and higher. In both scenarios, the error behavior of OPNET-MG had the same trends as the published results. In Scenario 2, the position error increases gradually and levels off or grows more slowly at range errors 0.07 to 0.12. With no range error in the network, OPNET-MG produces no position error. This shows the implementation is functional and able to correctly localize the network in the same way described in [LSS04a]. Inspection of the localization times of each node of several runs confirmed that OPNET-MG does indeed localize the network in an incremental fashion. In other

words, the area of localized nodes grows out from the starting triangle. The difference in behavior can be caused by one or more several possible factors. No original code or any supporting data (including confidence intervals) used to create and test the Map Growing algorithm was available for the development of OPNET-MG. Furthermore, the original Map Growing algorithm was implemented and tested in Matlab which is a much different development environment from OPNET. OPNET is a discrete event simulation environment, while Matlab is a matrix driven computation and simulation environment. The fundamental difference is that OPNET models the passage of time while Matlab does not. Since the only specification of the algorithm is the paper in which it was published, several details of the algorithm are vague and left to interpretation.

For example, at the start of the algorithm, [LSS04a] assumes each node knows the one-hop distance to each of its neighbors. In OPNET-MG, this is accomplished by an exchange of messages in which each node broadcasts a "HELLO" and responds to each received "HELLO". This enables each node to build a list of neighbors. Since, later in the algorithm, each node needs to know the neighbors of its neighbors, each node also broadcasts a message that includes its neighbor list. In [LSS04a], the starting node is required to have a degree no smaller than any of its neighbors, but is chosen randomly. In an effort to choose the best starting node possible, OPNET-MG implements a flood-based voting algorithm in which only messages originally from the node with the higher degree get forwarded on, until eventually all nodes are aware of the starting node. Once the starting node is picked, [LSS04a] chooses 2 nodes that form a triangle with no angle less than 30^o, and sets the coordinates for each of these nodes based on the one-hop distances between each pair of nodes and trigonometry. After the initial 3 nodes have their position, they broadcast it to all their neighbors. If another node receives 3 different position broadcasts it can use trilateration and the known one-hop distance estimate to localize itself. The trilateration method is not disclosed in [LSS04a]. OPNET-MG implements the trilateration method outlined in [LR03]. This method is described in detail in Appendix B. This method also

uses residue to signal an inconsistent result. Residue is essentially an average of the differences between the given one-hop distances and the distance resulting from the trilateration. The residue equation is also included in Appendix B. From empirical testing, it was found that rejecting trilateration results with a residue greater than $\frac{RANGE}{12}$ produced the best results. Thus, the algorithm could be selective and reject combinations of neighbor positions that would result in erroneous position estimates.

If a node does not have 3 localized neighbors, it uses the 2 beacon solver provided. First the node estimates the two possible positions it can be at based on its 2 localized neighbors. Now the node leverages the constraint that his localized neighbors have neighbors the node doesn't know about. For example, in Figure 4.3, P is able to determine his position is either at p_1 or p_2. Assuming the position broadcast messages from A and B also included a list of localized nodes that they know about, P can deduce that, since it does not know about N, its position cannot be p_1 and must be p_2. In practice, a node may have more than 3 localized neighbors but not be able to localize accurately with any combination of them, because the residual is too high. In this case, the node can use one of the many combinations of 2 nodes and the 2 beacon solver to estimate its position. The description of Map Growing in [LSS04a] did not specify a clear way to proceed in this situation. Based on pilot studies, it was found that the best results were obtained when a node averaged the results of all the combinations and then discarded any results far from the average. A

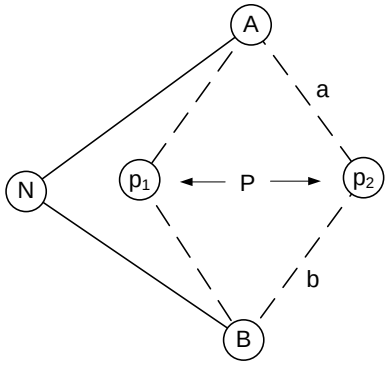


Figure 4.3: Two Beacon Solver: Nodes A, B, and N have localized. P uses information about N to eliminate p_1 or p_2 as a possible position

combination of ranging errors and geometry made it possible for the 2 beacon solver to actually give the wrong answer. Discarding results far from the average prevented these from being used in the estimate. The threshold at which to discard a result was empirically found to be $\frac{RANGE}{3}$.

The authors of Map Growing claim that the algorithm will localize all nodes because of a final phase that uses 3-6 of their closest neighbors and lateration. The percent OPNET-MG localized in both scenarios is plotted in Figure 4.4. This claim implies the resulting estimate is accepted no matter how high the residue value may be. In testing of OPNET-MG, this method provided localized nodes with very high position errors. This often happened when a node could not localize in a previous phase because its localized neighbors were collinear. Accepting the trilateration results of 3 collinear localized neighbors provides very poor position estimates. As a result, OPNET-MG only allows a node to localize in the final phase if it finds a set of three neighbors that trilaterates with a residue $< RANGE$. The result is that less than 100% of the nodes are localized, especially in high range error conditions, but it avoids the position error of the nodes localized with a high residue in the final phase.

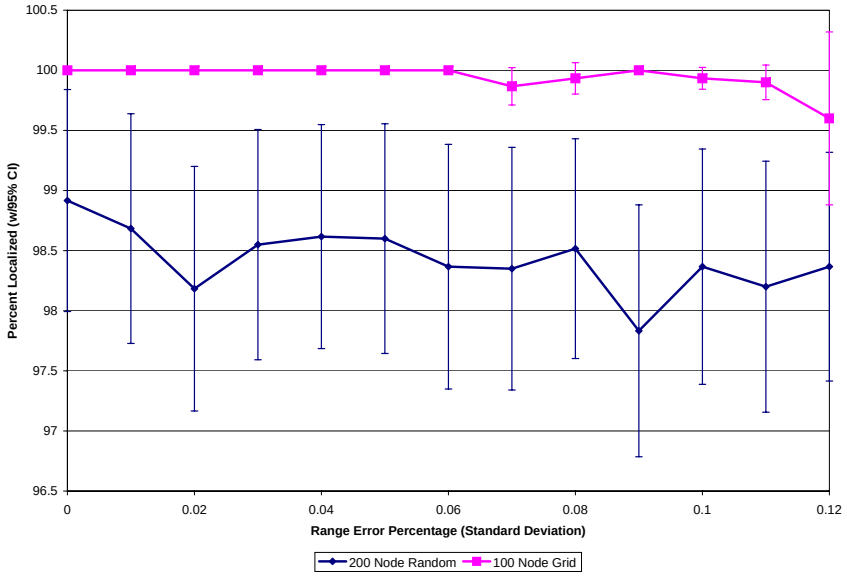


Figure 4.4: OPNET-MG Percent Localized by Range Error, both Scenarios

The final phase of Map Growing is only used if anchors are available and global coordinates are desired. During this phase, all anchors broadcast their global and relative coordinates. This information is relayed across the network until all nodes have information about all anchors. The nodes then use three of the anchors and a two-dimensional affine coordinate transformation to determine their global coordinates [WG97]. This phase is implemented for the purposes of verifying OPNET-MG, but it is not used in the actual experiments, since the focus of the work is on anchorless localization algorithms.

While the performance of the OPNET-MP did not exactly match the published results exactly, the major trends in its behavior performed similarly. Also, OPNET-MG was confirmed to localize the network in an incremental fashion. This meets the two major goals of the OPNET-MG and allows it to be compared against a concurrent localization algorithm.

4.3.2 AFL. The published AFL results [PBDT03] used several scenarios to test different aspects of the algorithm and to compare it to other types of algorithms. The scenario that most closely resembled the intended use of AFL herein is used to validate the OPNET version of AFL (referred to as OPNET-AFL when being compared to the published results). The scenario includes 10 repetitions on a 250 node network with fractional range errors ranging from 0.01 to 0.07 and the average degree of the network ranging from 5 to 13. The fractional error is modelled as described in [PBDT03]. According to [PBDT03], one-sided errors are more difficult to overcome, so the ranging error is taken as a sample from a uniform random distribution between zero and a fraction of the true distance between two nodes. For example, if the range error level is 0.01, and the true distance between two nodes is 10m, a value sampled from a uniform random distribution between 0 and 0.1m would be multiplied by the true distance, used as the range error and then simply added to the true distance. Even though this range error model is used for validation, the range error model described in the previous chapter is used in the actual experiments. The

metric used in [PBDT03] is Global Energy Ratio (GER). The intent of this metric is to capture the global structural property of the resulting network and report errors due to variation in range accuracy. GER is the root-mean-square normalized error value of the node-to-node distances [PBDT03]. GER is

$$\text{GER} = \frac{\sqrt{\sum_{i,j:i<j} \hat{e_{ij}}^2}}{N(N-1)/2} \tag{4.1}$$

Where e_{ij} is the difference between the true distance and the localized distance, and the error normalized by the true distance is $\hat{e_{ij}} = \frac{\hat{d_{ij}} - d_{ij}}{d_{ij}}$. Since the denominator includes $N(N-1)/2$ this metric reports disproportionately smaller GER values for larger networks. As such, it cannot be used to compare performance of networks of different sizes. However, to compare OPNET-AFL against its published results, GER is collected and reported. The performance of the OPNET-AFL and AFL's published results are shown in Figures 4.5 and 4.6. Two dimensional versions of these graphs split up by degree are included in Appendix C. Examining the graphs and their source

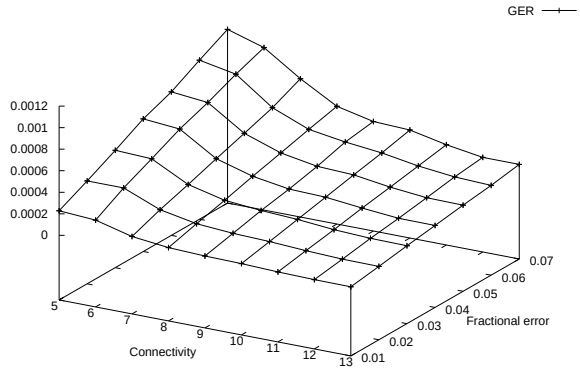


Figure 4.5: AFL Published GER [PBDT03]

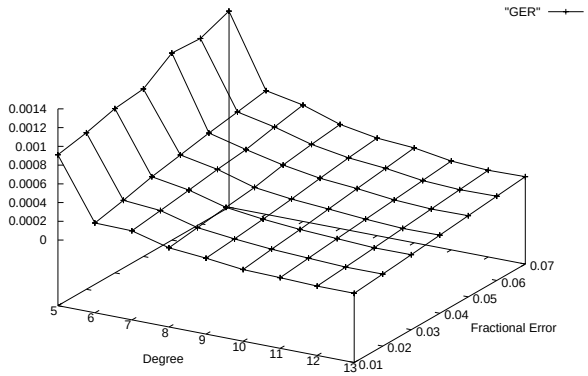


Figure 4.6: OPNET-AFL GER

data shows that OPNET-AFL performs similarly and often better than the baseline version with degrees 6 and above. In both versions, GER increases as connectivity drops and as fractional error increases. The performance numbers are also very close. The only major difference is with networks of degree 5. OPNET-AFL performs much worse in this case. If a graph of the network is considered a bar-and-joint framework,

it is rigid if it cannot be flexed while preserving the distances (as in a rectangle, for example). Even if the graph is rigid, it may be subject to local flips. For example, if there are just two triangles sharing an edge, one triangle can be reflected through that edge without any distances changing. We call such a graph rigid but not globally rigid. For AFL to work given just edge lengths, we need a globally rigid graph that has exactly one embedding [PBDT03]. The AFL authors report necessary precautions were taken to reduce the possibility of non-rigid graphs, which is very important with low degree networks. A uniform local density was maintained by adding nodes to those positions that had a number of neighbors below a certain threshold based on the average degree of the network. Despite requesting details about this method of network generation, the actual method was not available to replicate. Instead, the constant density network generation method was used, in which the network area is split into a grid and an equal number of nodes are randomly placed in each square. This method seems to be sufficient for degrees 6 and higher, but for degree 5 it did not effectively prevent non-rigid graphs resulting in poor performance on those networks. Upon inspection of the GER values for the degree 5 experiments, 3 samples are found to be consistently 2-3 times worse than the others in the set for all fractional range errors. These networks are likely non-rigid graphs and drive the GER values up, while also increasing the 95% confidence interval to 30-60% of the value. Fortunately, since networks with an average degree less than 8 are not used in this research, this is not a critical issue.

As with the Map Growing algorithm, some details of AFL are not explicitly defined. One instance of this occurs in the first phase of the algorithm. During this phase, an arbitrary node n_0 uses hop counts ($h_{0,1}$ is the number of hops from node n_1 to n_0) to determine which node, n_1, is farthest away. Node n_2 is chosen by finding the node that maximizes $h_{1,2}$. Node n_3 minimizes $|h_{1,3}-h_{2,3}|$ while maximizing $h_{1,3}+h_{2,3}$. This causes n_3 to be roughly equidistant to n_1 and n_2 but also on or near the edge of the network. Node n_4 is similarly found as the node that minimizes $|h_{1,4} - h_{2,4}|$ and maximizes $h_{3,4}$. At this point, nodes 1-4 are on the edge of the graph and form a

set of axes that are roughly perpendicular. Node n_5 is the approximate center of the network and is found by minimizing $|h_{1,5} - h_{2,5}|$ and minimizing $|h_{3,5} - h_{4,5}|$. Once all 5 reference nodes are found, and all nodes know their hopcount to each, each node can estimate its position using polar coordinates. In OPNET-AFL, this is accomplished with 5 rounds of a node requesting all others to send it their hopcounts. Care was taken to prevent repeated traffic if possible. For instance, subsequent copies of the same request are ignored by a node.

A node uses polar coordinates converted to cartesian coordinates as its initial position. Once it has its initial position, the node broadcasts its position to its neighbors. The description of AFL in [PBDT03] specifies each node will periodically send this position estimate out to all its neighbors. A node is to estimate its position based on its current estimated position, the measured distances between itself and its neighbors and the estimated positions of its neighbors. However, the timing of broadcasting a node's position estimate, receiving estimates from all of its neighbors, and estimating a new position was not described in [PBDT03]. In OPNET-AFL, a node updated its position every 5 seconds and sent out a broadcast of this new position immediately after. This seemed to allow enough time for a node to collect enough updates to make the effort of a new estimate worth it, while not wasting too much time waiting for updates. When it was time to estimate a new position, a node n_i calculates the estimated distance $\hat{d}_{i,j}$ to each neighbor n_j. It already has the measured distance, $r_{i,j}$ to each neighbor from the first phase. Let $\hat{v}_{i,j}$ represent the unit vector in the direction of the estimated position of n_i to the estimated position of n_j. The force $\overrightarrow{F}_{i,j}$ in the direction $\hat{v}_{i,j}$ is

$$\overrightarrow{F}_{i,j} = \hat{v}_{i,j}(\hat{d}_{i,j} - r_{i,j}). \tag{4.2}$$

The sum of the forces from all neighbors gives the resulting force on the node n_i

$$\overrightarrow{F}_i = \sum_{i,j} \overrightarrow{F}_{i,j}. \tag{4.3}$$

The difference between the measured and estimated distances creates a corresponding position energy $E_{i,j}$ and has a magnitude of $\overrightarrow{F}^2_{i,j}$. The total position energy of a node n_i is

$$E_i = \sum_j E_{i,j} = \sum_j (\hat{d}_{i,j} - r_{i,j})^2. \tag{4.4}$$

The position energy of each node reduces when it moves an infinitesimal distance in the direction of resultant force. The distance the node moves should satisfy two conditions. 1) The energy of the new position must be less than the energy of the current position. 2) Moving to the new position does not result in a local minima. Since the energy at the current and new position can be calculated, the algorithm can guarantee the first condition is met. The second condition is met by moving only small amounts inversely proportional to the number of neighbors $\frac{|\overrightarrow{F}_i|}{2m_i}$. This value was empirically selected by [PBDT03] and is used in OPNET-AFL. The last part of the algorithm determines how a node decides to stop estimating new positions which is not explained in [PBDT03]. Early testing of OPNET-AFL, indicated a simple threshold for the move distance did not perform consistently in all network sizes and all degrees. Monitoring of the position energy of the nodes reveals the behavior of the refinement. As can be seen in Figure 4.7, the position energy starts very high because the phase one position estimate is quite inaccurate. The position energy drops rapidly, then levels off, and drops more slowly. At first glance it seems this supports the usage of a simple threshold for the change in energy. Unfortunately, different nodes bottom out at different levels, Additionally, this position energy also depends on node degree. Figure 4.8, shows the energy for a node through the end of the run. A very low move distance threshold is used in Figure 4.8 to observe the behavior over time. From this figure and observing similar plots from other nodes, it is evident that the energy eventually rises slowly. This is possible because the neighbors of the observed node are still updating their positions causing the energy of the observed node to change. Through testing it is known that this period also signals the end of

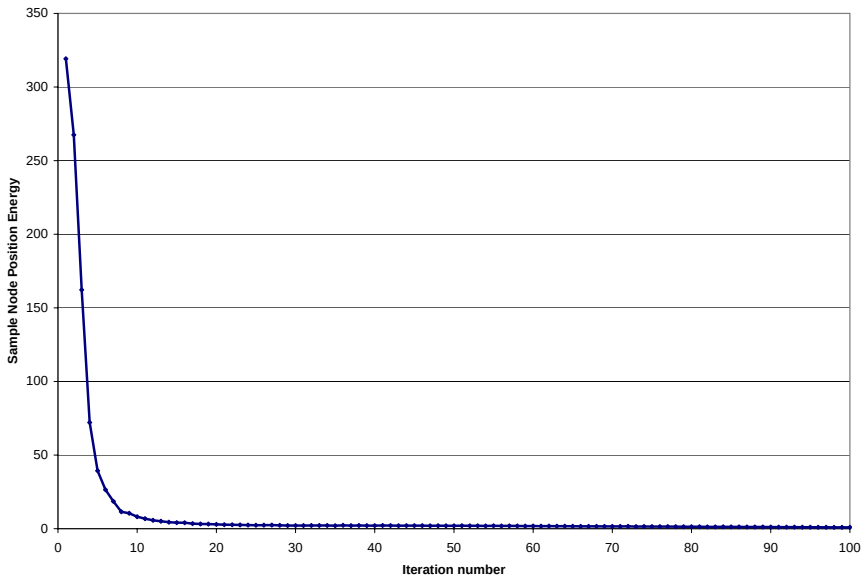


Figure 4.7: Position Energy of a Sample AFL Node vs First 100 iterations

useful iterations. As a result OPNET-AFL maintains a count of the current number of iterations completed and an average of the previous window of position energy values for that node. Once a new average surpasses a previous average, the node can stop estimating positions. An average is used to ensure the general trend is moving up. The size of the window depends on the degree of the node.

The verification of OPNET-AFL shows it behaves very closely to or better than the published results in almost every situation. The only situation that it does not match closely is with networks with an average degree of 5 due to differences in the network generation method. Like the Map Growing algorithm, several implementation decisions had to be made in the absence of an explicit description of the algorithm details.

4.4 Data Collection Methods

Using OPNET implemented versions of an incremental (Map Growing) and concurrent (AFL) wireless sensor network localization algorithm, the experiment seeks to determine 1) the most important factors to energy consumption in localization, 2) the type of anchorless algorithms that provide the best position accuracy and the most

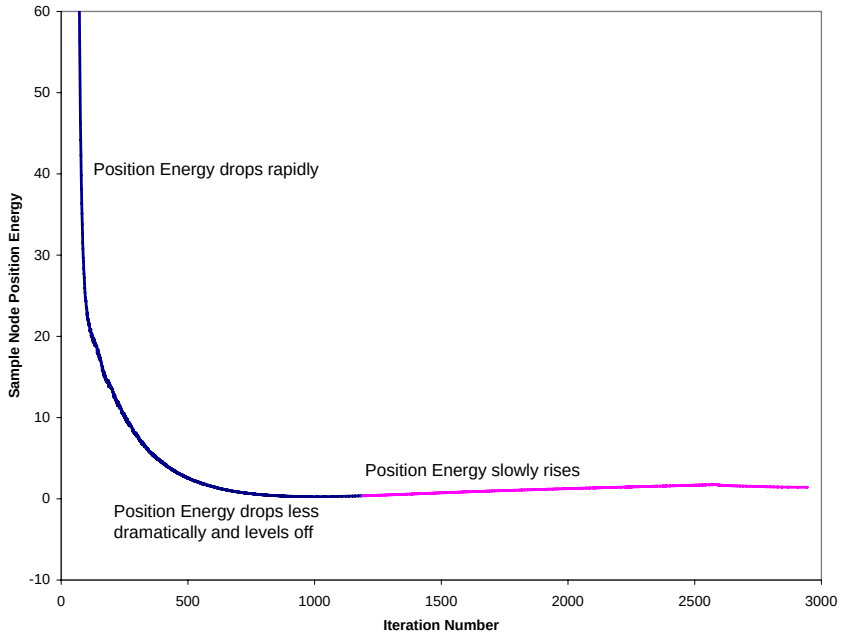


Figure 4.8: Position Energy of a Sample AFL Node through the end of the run

efficient energy usage and 3) to develop an energy consumption model to predict energy consumption of incremental and concurrent algorithms in various networks environments. To this end, the localization models are run on a set of network of differing types (Constant Density and Random Uniform), differing sizes (30, 100 and 300), differing average degrees (8, 12 and 16) under differing range error conditions (normal distributed errors averaging 2, 5 and 10% of the actual distance). Networks are created by using Perl scripts, listed in Appendix D. The Perl scripts read in a list of seed numbers then generate a network for each seed varying the size, degree and type for all 18 (2 types, 3 sizes, 3 target degrees) desired combinations. The scripts check that the graph is connected and is within the desired range of the target degree. Networks in this experiment have an average degree in the range of the target degree ± 0.05. Data is collected by text output from the OPNET model that reports the factors and other data on a per node and per run basis. The data is then aggregated and analyzed. For the average transmitted and received bits, the OPNET model implementing the algorithm also attributed a number of bits associated with every field in every packet sent and received through the end of the simulation. The

number of bits and messages sent and received is totaled and averaged over all nodes in the network, yielding the average number of bits received and transmitted for that run. The average distance error is calculated at the end of the simulation after every node has either localized or finished trying. The percent localized is calculated and written to the output file at this time also. In all graphs and tables in this section, a confidence interval of 90% is used.

4.5 Error Performance

The ultimate goal of any localization algorithm is to accurately determine the position of all the nodes in the network. For both AFL and Map Growing the average distance error performance is analyzed with respect to each factor. A computation of effects is also used to determine how the levels of each factor affect accuracy. Additionally, Appendix E includes graphs of the networks on which AFL and Map Growing achieved their best, worst and average performance. Graphs of the resulting position estimates from both algorithms are also included for comparison. This should provide a qualitative assessment of the performance for each.

4.5.1 Algorithm. Since AFL uses significant refinement in its position estimation, it is expected to produce more accurate localization results. As shown, in Figure 4.9, AFL meets expectations. In both CD and RU networks, AFL has a much lower resulting Average Distance Error (ADE) than Map Growing. On the average, with CD networks the ADE of Map Growing is well over twice that of AFL. For RU networks, they are closer but AFL is still much better due to the extensive refinement that AFL does. Map Growing has higher errors for several reasons. Map Growing is an incremental algorithm that starts with 3 localized nodes and expands to the edges of the network. The accuracy of every node's position estimate is dependent on the accuracy of the two or three neighbors that nodes uses to localize with. These conditions allow errors in earlier position estimates to be carried into estimates made

by nodes that localize later. As a result, errors often grow the farther a node is from the starting triangle. With no refinement, these errors remain in the final estimate.

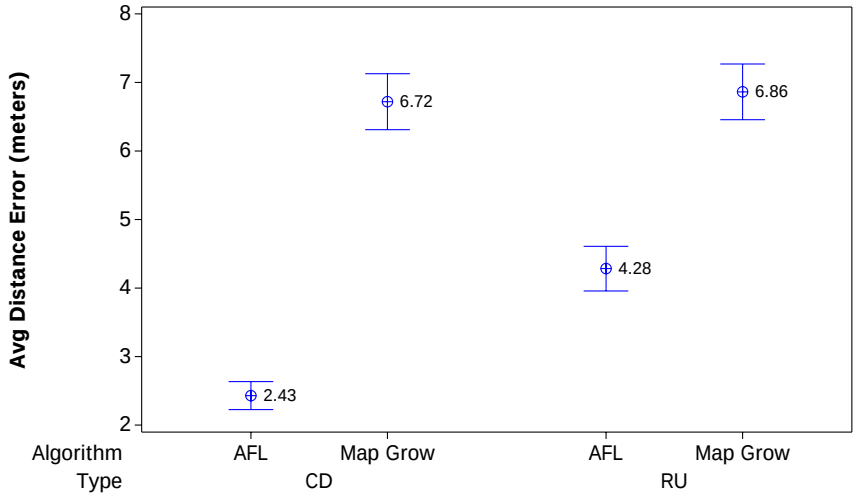


Figure 4.9: Plot of Average Distance Error by Algorithm and Network Type

4.5.2 Network Type. While Map Growing performs statistically no different on CD and RU networks (according to Table H.8 in Appendix H), AFL has a significantly higher ADE for RU networks. CD networks are more evenly spread out over the network area and all nodes tend to have very similar degrees. RU networks are randomly deployed and often have areas with much higher degrees and other areas with much lower degrees. Since Map Growing only needs two or three eligible nodes to localize with, the degree is less important. Additional neighbors do not help at all. AFL is different. Each node uses input from all its neighbors to refine is position and the more neighbors it has, the more position "force" corrections it can use. Since RU networks have areas with lower degrees, these nodes suffer from having less neighbors and result in worse position estimates. CD networks have less of these type of nodes, so AFL is able to localize them all equally well. Similarly, when using AFL, nodes are better served by having their neighbors spread around the node evenly, rather on one side only. This creates more directions from which the position corrections are provided. Nodes that have neighbors only on one side do not have this benefit and their position accuracy suffers. If a network is more concave, or curves inward, in

many places, more nodes fall in this category. With fewer directions to help balance the position "forces", the node requires more iterations and achieves less accuracy. This phenomenon is confirmed by comparing ADE performance with images of the network layout such as those in Appendix G, which has examples of concave and convex networks. There are more RU networks that have this behavior than CD networks. This is another reason AFL performs worse on RU networks. On the other hand, Map Growing only depends on the direction to two or three of its neighbors, so it is less susceptible to this effect.

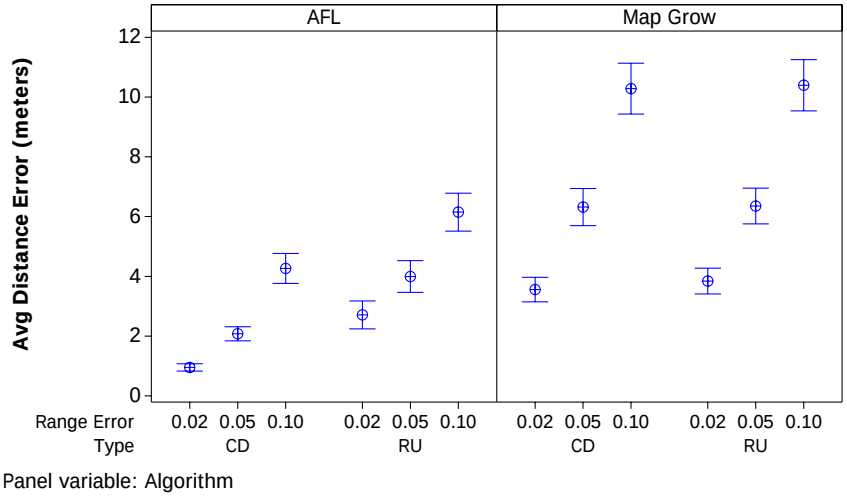


Figure 4.10: Average Distance Error vs Network Type and Range Error

4.5.3 Range Error. For most localization algorithms, range error negatively affects the accuracy of the position estimates. As seen in Figure 4.10, this is also the case with AFL and Map Growing. Both algorithms perform significantly worse as range error is increased. The only difference between the two algorithms is that the ADE of Map Growing degrades much more quickly than AFL. This is also probably due to the refinement AFL does. Map Growing accepts the distance estimate with all the range error it may have, while AFL iteratively refines the position estimate and reduces the effect of errors in the distance estimate.

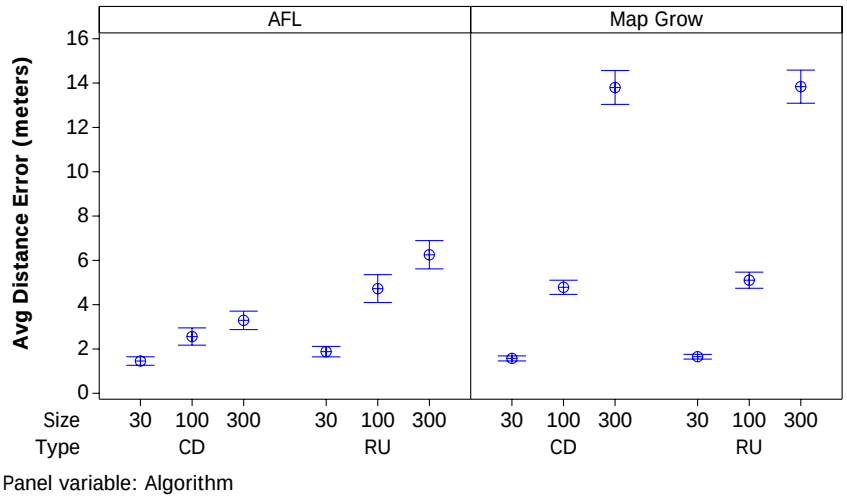


Figure 4.11: Average Distance Error vs Network Type and Size

4.5.4 Network Size. Examining how ADE changes with network size gives an indication of how well a localization algorithm scales. The ADE of both AFL and Map Growing versus Network Size and Type is shown in Figure 4.11. The figure suggests that Map Growing does not scale nearly as well as AFL. As size increases for Map Growing, the ADE increases very quickly as compared to AFL. Also, for AFL as the network size grows from 100 to 300, the ADE levels off and does not increase as much. The ADE for Map Growing increases by large amounts for increases in the size. For instance, the ADE for Map Growing with 300 nodes is about double the ADE of AFL in RU networks and more than triple in CD networks. Map Growing allows errors in previous estimates to be carried forward to degrade future estimates of other nodes. If a network is larger, this propagation of errors is carried further. The further it is carried, the worse the error gets until the map grows to the edge of the network.

4.5.5 Degree. AFL depends heavily on node degree for its refinement. A node will localize quicker and more accurately with more neighbors. On the other hand, Map Growing only depends on having two or three eligible neighbors. Anything beyond that does not make much of a difference in terms of accuracy. These trends are both illustrated in Figure 4.12. The figure shows that the ADE of AFL drops

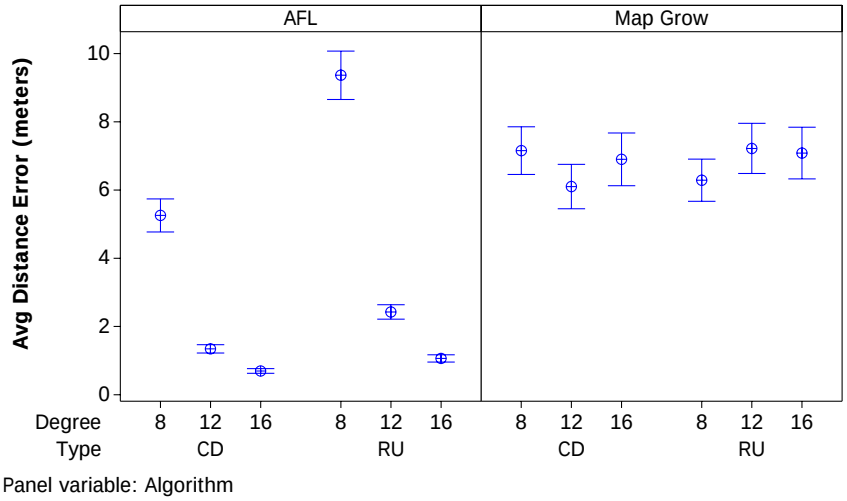


Figure 4.12: Average Distance Error vs Network Type and Degree

dramatically as the degree is increased. ADE is best when the degree is highest. According to table H.8 in Appendix H, which is the table of effects for ADE using Map Growing, the effect of degree 8 and 12 is not significantly different. Also, the effect of degree 16 (± 0.2) is very slight. This indicates the accuracy of Map Growing does not change very much as the average degree of the network changes. While increasing the degree of the network when using AFL will improve ADE, there are diminishing returns above degree 12. The amount of improvement between degree 8 and 12 is much more than between 12 and 16. This is true for both CD and RU networks. This is likely because once a node has a certain number of neighbors and is able to localize effectively, more neighbors do not help as much.

4.5.6 Computation of Effects on Average Distance Error. A computation of effects determines the effect each level of a factor has on a response. Figures 4.13 and 4.14 plot the main effects of each level of each factor for ADE. The AFL main effects plot shows how ADE increases with network size but increases more slowly after 100 nodes. Increasing the degree reduces ADE but levels off at higher degrees. AFL performs better on CD networks than RU because of the more stable degree and less incidence of concave areas in CD networks. Range error degrades ADE for AFL in an approximately linear fashion. Figure 4.14 shows how poorly Map Growing

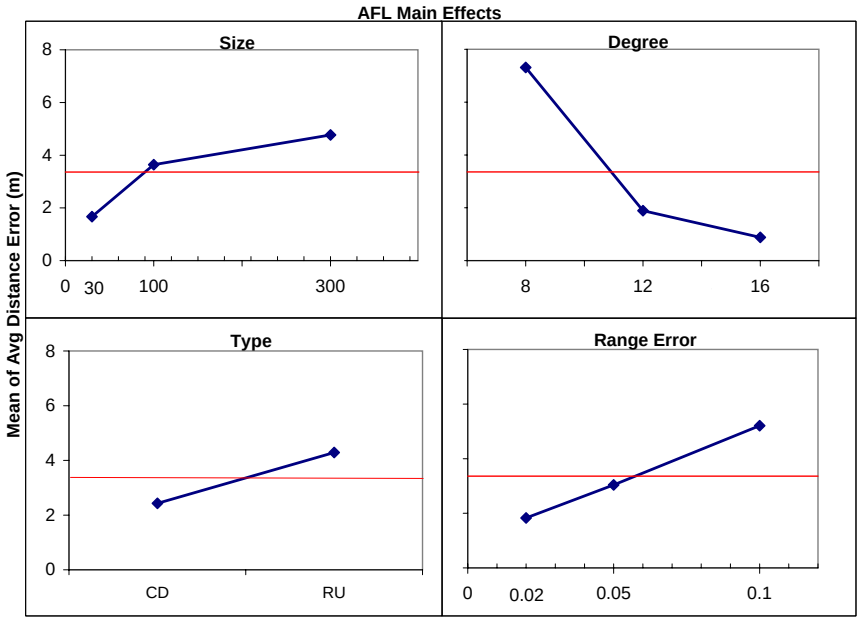


Figure 4.13: AFL Main Effects Plot of Average Distance Error

scales in terms of accuracy. It also shows how degree and type have little effect on Map Growing position accuracy. Similar to AFL, increasing the range error results in a linear increase in ADE. The ADE Computation of Effects table for both Map

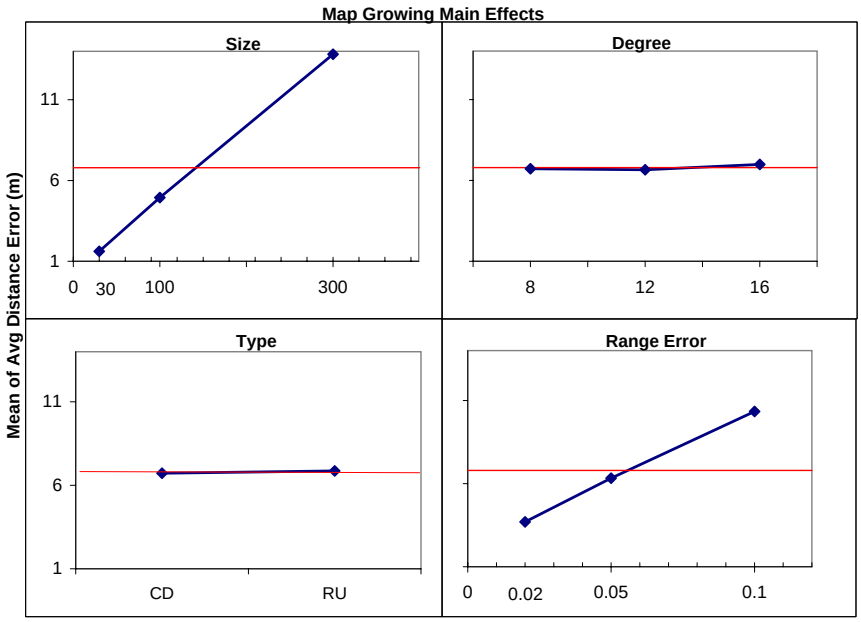


Figure 4.14: Map Growing Main Effects Plot of Average Distance Error

Table 4.1: Table of Effects for Map Growing Average Distance Error

Parameter	Level	Mean Effect	Std Dev	Upper CL	Lower CL
Mean		6.79	0.08	6.65	6.93
RE	0.02	-3.09	0.12	-3.29	-2.89
	0.05	-0.46	0.12	-0.65	-0.26
	0.10	3.55	0.12	3.35	3.74
Degree	8	-0.07	0.12	-0.27	0.13
	12	-0.13	0.12	-0.33	0.07
	16	0.20	0.12	0.00	0.40
Size	30	-5.18	0.12	-5.38	-4.98
	100	-1.85	0.12	-2.04	-1.65
	300	7.03	0.12	6.83	7.23
Type	CD	-0.07	0.08	-0.21	0.07
	RU	0.07	0.08	-0.07	0.21

Growing and AFL is included in Appendix H. A summary of the ADE effects is shown in Tables 4.1 and 4.2. The tables show that Map Growing network size has the largest effect on the mean followed by range error. It also shows that neither degree 8, 12 nor 16 are significantly different than the mean. For AFL, the tables show degree has the largest effect on the mean followed by range error and network size, which have similar effects. All levels of all factors for the ADE of AFL are significantly different than the mean. Tables for the contrasts of each pair of levels for all factors and both algorithms are included in Tables H.9 and H.18 in Appendix H. At a 90% confidence level, the contrasts reveal there is no statistical difference in ADE between degree 8 and 12 and either type for Map Growing. For AFL, all levels of all factors are significant at the 90% confidence level. This suggests Map Growing performs equally well in terms of accuracy on networks of either type, with degrees 8 or 12.

4.6 Percent Localized

The percentage localized metric indicates how many of the nodes an algorithm is expected to localize under different conditions. The better coverage an algorithm has, the more likely the localization results can be used for sensor coordinates or location based routing.

Table 4.2: Table of Effects for AFL Average Distance Error

Parameter	Level	Mean Effect	Std Dev	Upper CL	Lower CL
Mean		3.36	0.06	3.25	3.46
RE	0.02	-1.53	0.09	-1.68	-1.38
	0.05	-0.32	0.09	-0.47	-0.17
	0.10	1.85	0.09	1.70	2.00
Degree	8	3.95	0.09	3.80	4.10
	12	-1.47	0.09	-1.62	-1.32
	16	-2.48	0.09	-2.63	-2.33
Size	30	-1.69	0.09	-1.84	-1.54
	100	0.28	0.09	0.13	0.43
	300	1.41	0.09	1.26	1.56
Type	CD	-0.93	0.06	-1.03	-0.82
	RU	0.93	0.06	0.82	1.03

4.6.1 Algorithm. For AFL, any node connected to the network will receive all hopcount requests from the 5 reference nodes, since the requests are flooded. If all hopcount requests are received, the node is able to estimate a position using the polar coordinates described earlier. Once it has this position estimate, it is technically considered localized. Even if it has only one neighbor, it can still refine the position considerably. Since connectivity is all that is needed for a node to localize in AFL and since all networks in this research are connected, AFL localized all nodes under all configurations. This is shown in Figure 4.15.

Map Growing has different localization requirements. To 3-beacon localize, a node must have at least 3 neighbors that are non-collinear with a sufficiently small residual. The residual is a measure of the difference between the localized distances and the true distances. If there is error in the estimated distance (such as with RSSI) this increases the chance of a poor residual. For 2-beacon localization a node must have 2 neighbors that are non-collinear with the node and one must have a localized neighbor. Given these requirements, a low degree, error in the distance estimates, or poor geometry of nodes can all cause a node to not localize. This effect is reflected in Figure 4.15. Map Growing overall localizes 90% or more of the networks, but does not consistently localize 100% as AFL does. Since AFL networks localized all networks,

the main effect plots and table of effects are trivial and not included. The table of computation of effects, effects summary and contrasts for the AFL percent localized are included in Appendix H.

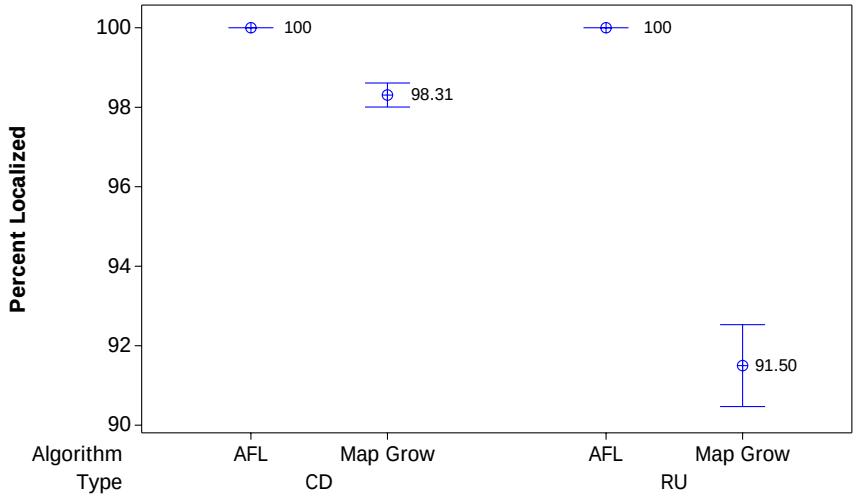


Figure 4.15: Plot of Percent Localized by Algorithm and Network Type

4.6.2 Network Type. Figure 4.15 also shows Map Growing does not localize RU networks as well as CD networks. This is likely due to RU's inconsistent degree across the network. Some areas of the network do not have enough nodes or not enough eligible nodes to localize with. Furthermore, with lower degrees it is possible a node that connects 2 components will not localize. In this case the component not yet localized may never localize, since the map cannot grow to it. For these reasons, Map Growing does not localize as many nodes on RU networks compared to CD networks.

4.6.3 Range Error. Figure 4.16 shows that range error has little effect on the percent localized using Map Growing. The average for a range error of 0.1 is slightly lower for both CD and RU networks, but it is within the 90% confidence interval of 0.05 in both cases. While range error may cause some nodes to not localize due to bad resulting residuals, this case is rare and does not affect Map Growing percent localized nearly as much as other factors.

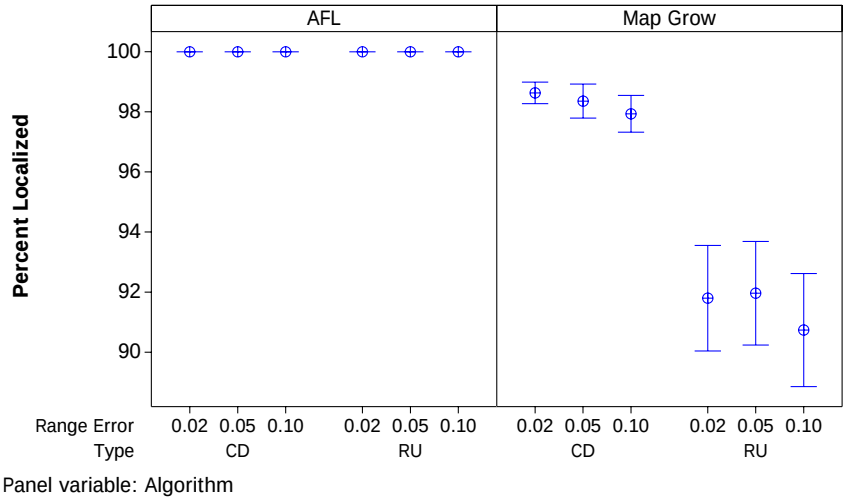


Figure 4.16: Percent Localized vs Network Type and Range Error

4.6.4 Network Size. For CD networks, network size does not affect the percent localized nearly as much as it does in RU networks as shown in Figure 4.17. RU 100 and 300 node networks perform significantly worse than 30 node networks, while all CD networks perform similarly. This is likely due to map partitioning. When two components of a network are connected by very few nodes, and those nodes do not localize, the entire unreached component is not localized. This effect is more probable as the network grows. Additionally, with an inconsistent degree across the network, as with RU networks, this effect becomes very common.

4.6.5 Degree. Figure 4.18 shows that in both CD and RU networks, Map Growing localizes significantly less with degree 8 networks, while achieving close to 100% with degree 12 and 16. In degree 8 networks, the partitioning effect discussed above is much more common. As a result, the percentage localized drops.

4.6.6 Computation of Effects on Percent Localized. Figure 4.19 plots the main effects of all factors on Map Growing percent localized. Range error is the only factor with little effect on percent localized. Degree has the largest effect on percent localized and degree 8 networks have the worst percent localized of any other single category. This plot also shows that the worst type of network for Map Growing is

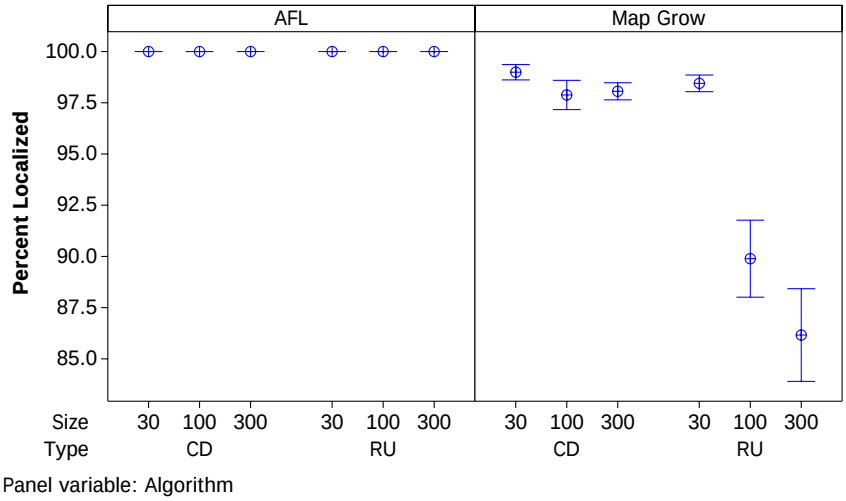


Figure 4.17: Percent Localized vs Network Type and Size

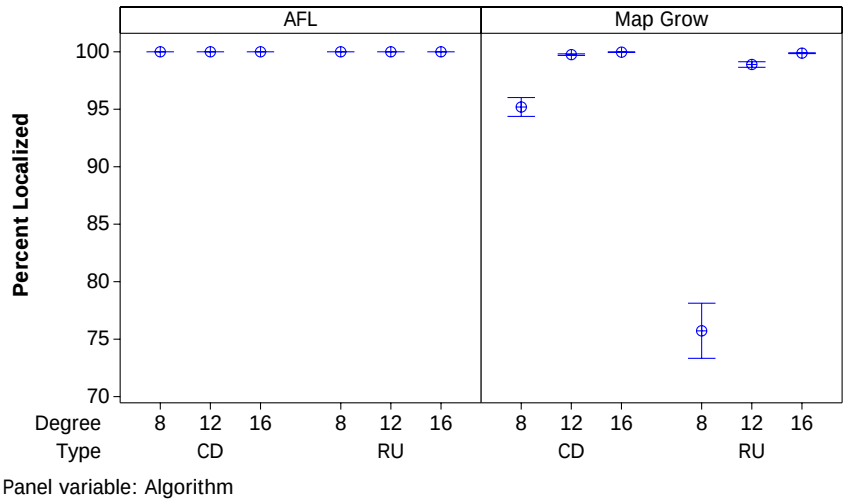


Figure 4.18: Percent Localized vs Network Type and Degree

RU, with 100 or 300 nodes and degree 8. This is the worst category of networks for Map Growing by far. The average percent localized for this category is 65.85%, while all other networks average 98.54%. With so few nodes localizing in this category, the effect can be seen in other responses. There are fewer nodes broadcasting their position and localized neighbors and the broadcasts that do occur are smaller. This drives down the number of messages and bits transmitted and received. Since so few nodes localized in this category and the behavior is drastically different than networks that localize more than 90%, the residuals of this category are not normally

distributed. This prevents the ANOVA and linear regression assumptions from being met. As a result, degree 8 networks are removed from the Average Transmitted and Received Bits ANOVA and regression. The table of effects for Map Growing percent localized is included in Table 4.3.

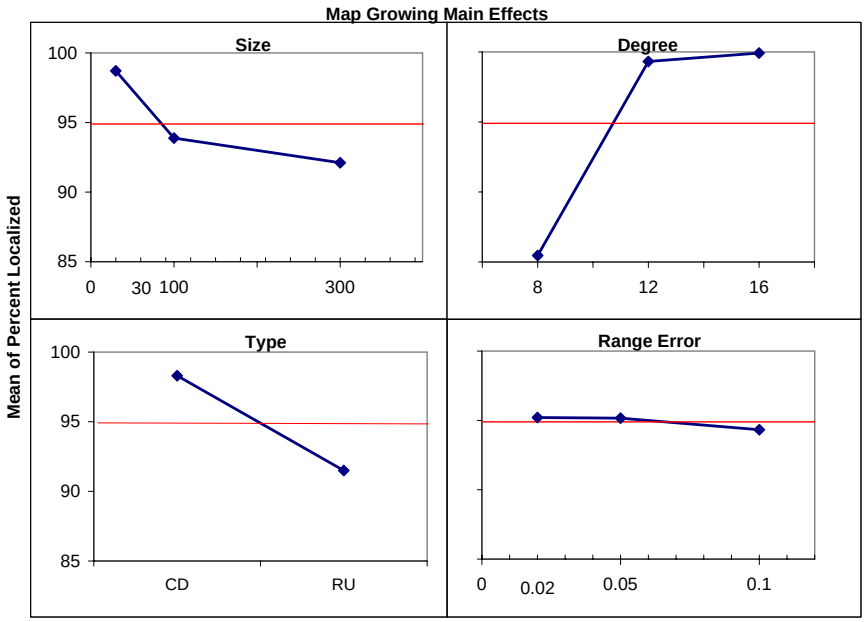


Figure 4.19: Map Growing Main Effects Plot of Percent Localized

Table 4.3: Table of Effects for Map Growing Percent Localized

Parameter	Level	Mean Effect	Std Dev	Upper CL	Lower CL
Mean		94.90	0.21	94.56	95.25
	0.02	0.31	0.30	-0.18	0.80
RE	0.05	0.26	0.30	-0.23	0.75
	0.10	-0.57	0.30	-1.06	-0.08
	8	-9.44	0.30	-9.93	-8.95
Degree	12	4.42	0.30	3.93	4.91
	16	5.02	0.30	4.53	5.51
	30	3.81	0.30	3.32	4.30
Size	100	-1.02	0.30	-1.51	-0.53
	300	-2.79	0.30	-3.28	-2.30
Type	CD	3.40	0.21	3.06	3.75
	RU	-3.40	0.21	-3.75	-3.06

4.7 Energy Performance

Since communication is the most expensive operation of a WSN, the number of bits transmitted and received is important to the overall energy consumption of a node. The following section analyzes the energy performance of each algorithm with respect to each factor. An ANOVA and computation of effects is presented to determine which factors contribute most to the energy variation and how much each level of the factors affects energy use.

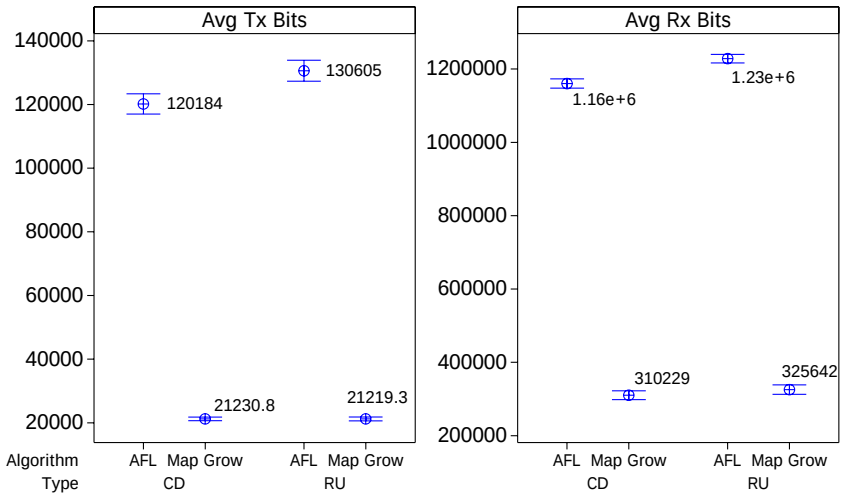


Figure 4.20: Average Transmitted & Received Bits by Algorithm & Network Type

4.7.1 Algorithm. To compare the communication cost of each algorithm, the average performance of average bits transmitted and received is plotted in Figure 4.20. The figure shows that AFL requires much more communication than Map Growing. In fact, on average AFL requires over 5 times as many transmitted bits and over 36 times as many received bits. The major reason for this difference is the AFL refinement phase which takes many iterations. For a typical network of any appreciable size, the AFL refinement can take between 700 and 2,500 iterations depending on the degree of the network. During each iteration, each node broadcasts its position and receives position broadcasts from all of its neighbors. As a comparison, Map Growing averages 35.43 messages for the entire algorithm. Additionally, during AFL's first phase to find the 5 reference nodes, all nodes are required to respond to a hopcount request from

one of the reference nodes 5 times. Each time, the request is flooded to all nodes in the network and every reply from all nodes is flooded back to the reference node. This phase is also expensive compared to Map Growing.

4.7.2 Network Type. Figure 4.20 also illustrates the average number of transmitted and received bits for each network type. In both, average bits transmitted and average bits received, Map Growing performed similarly in both the RU and CD networks. As shown in Table H.3 in Appendix H, the Map Growing Average Transmitted Bits is not statistically different for either type of network. At the same time, AFL is more sensitive to the difference and performs worse on RU networks than CD networks. Map Growing performs similarly on each network because the algorithm does not depend as much on the degree of each node. The algorithm depends more on the geometry of a node's neighbor. For Map Growing, having 8 neighbors is the same as 16 if no subset of 3 are non-collinear. On the other hand, AFL uses a "force" correction from each neighbor to refine the position. The more neighbors, the more information for refinement. The low degree areas of the RU networks have less information to refine with, so they work longer with more messages to balance the "force" corrections. As a result, AFL requires more messages for RU networks, causing more bits to be sent and received.

4.7.3 Range Error. Since range errors cause range aware algorithms to localize on less accurate information, range errors are expected to cause the number localization criteria fail more often forcing algorithms to take longer to localize and use more energy in the process. As Figures 4.21 and 4.22 show, the communication of both AFL and Map Growing is affected little by varying range errors. In the Map Growing plots, for both Average Received and Transmitted Bits, the mean at each range error level is within the confidence interval of the other range error levels, which means that the means of each range error level are not statistically different from each other. Since the confidence intervals of all the range errors also contain the mean, the range error factor does not have a significant effect on either Average Received or

Transmitted Bits. It turns out that range errors only causes the localization criteria to fail in a few cases for Map Growing. Similarly, the mean of the Average Transmitted and Received Bits at each range error level is within the mean of the other levels, indicating that each range error level is not statistically different from each other and that range error is not a significant factor for AFL communication also.

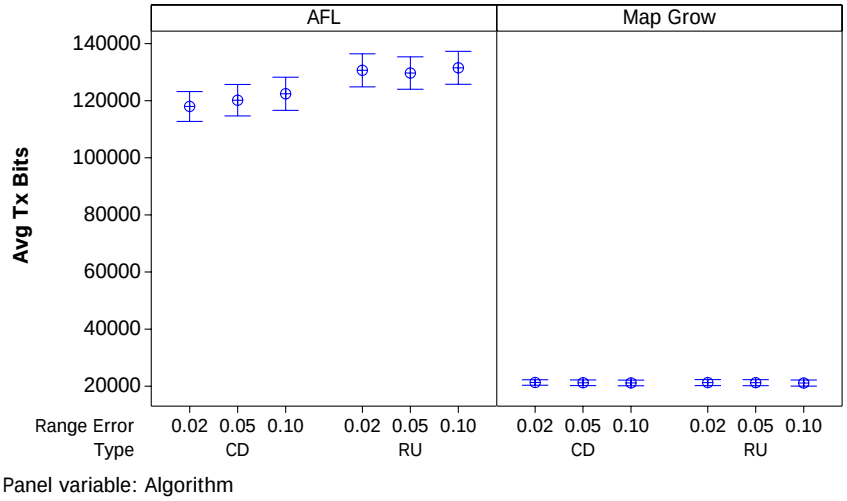


Figure 4.21: Average Transmitted Bits vs Network Type and Range Error

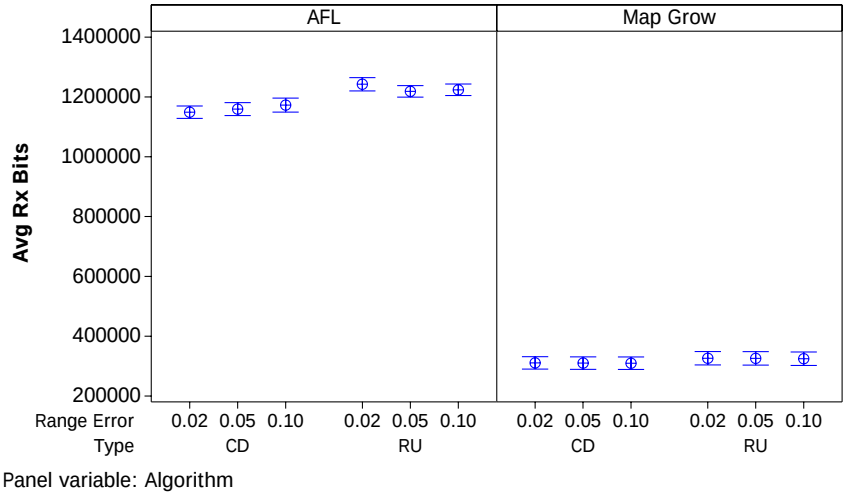


Figure 4.22: Average Transmitted Bits vs Network Type and Range Error

4.7.4 Network Size. Similar to range error, the size of the network does not change the average communication of Map Growing. In Figure 4.23 and 4.24,

the different levels of the network size factor are not significantly different from each other, implying that the Average Transmitted and Received Bits does not change significantly in response to changes in the network size. This is true because with Map Growing there is little network-wide traffic that causes the average node communication to increase. On the initial starting node vote requires messages to traverse the network. The rest of the algorithm is dependent on the immediate neighborhood of the node. The size of the network would not change the characteristics of this immediate neighborhood and therefore has little effect on average node communication costs. As for AFL, an increase in the size of the network slightly increases the average number of transmitted and received bits as shown in Figures 4.23 and 4.24. Unlike Map Growing, AFL has several steps in the algorithm that require network wide communication. To find all 5 reference nodes a request is flooded to all nodes and a response from all nodes is flooded back to the requester. With more nodes in the network a single node is likely required to forward more replies, increasing that node's number of transmits and receives. Since the reference node finding phase uses much less communication than the refinement phase, this phenomenon has only a modest effect on the average transmitted and received bits. The refinement phase does not depend on the size of the network and also accounts for most (well over 90%) of the transmitted and received bits in AFL. In Random Uniform networks, AFL transmitted and received bits increases with network size. Since RU have areas with differing node degrees, they are more likely to have areas that are connected by only one or two links between nodes. This creates a "bottleneck" path for network wide communication. Nodes along this path transmit and receive an inordinate amount because of the extra traffic to forward. RU networks are randomly deployed so there is no guarantee that there will be any nodes in a given area. As a result, RU networks are more likely not to have a direct path between nodes. As a result, the network wide traffic has to traverse around areas with no nodes increasing the communication used. CD networks avoid both of these issues by having a relatively constant degree across the network.

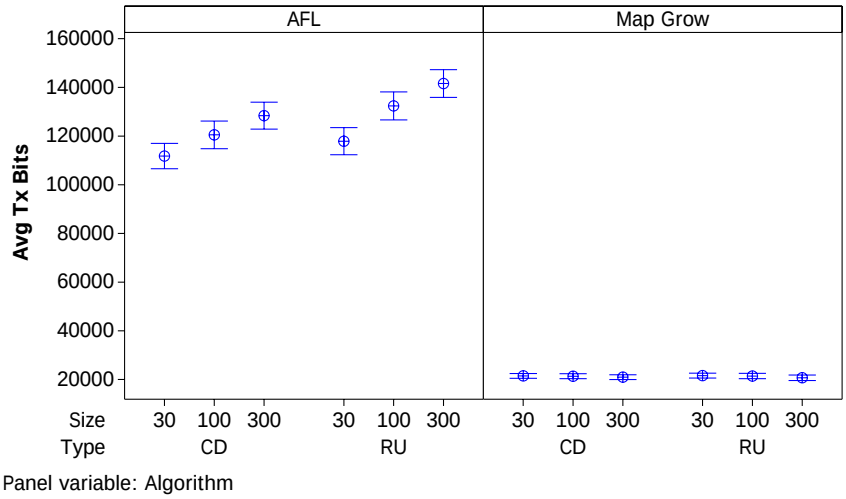


Figure 4.23: Average Transmitted Bits vs Network Type and Size

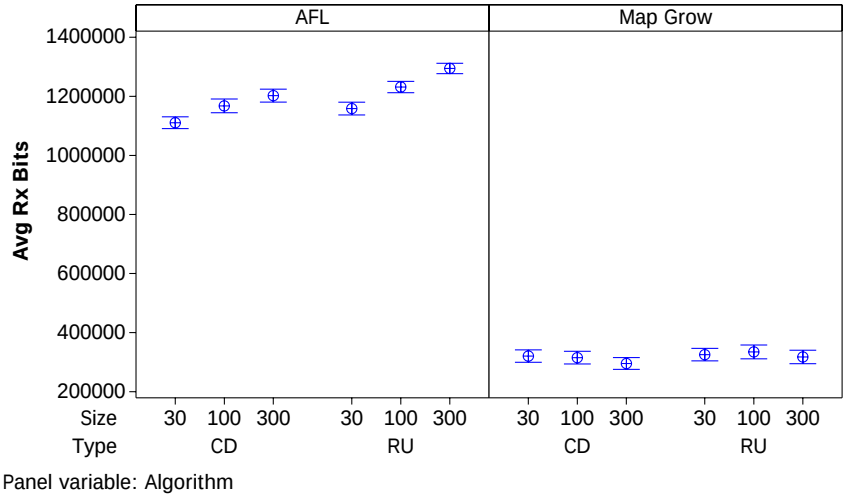


Figure 4.24: Average Received Bits vs Network Type and Size

4.7.5 Degree. Unlike range error and network size, degree has a very significant effect on both Map Growing and AFL communication, as can be seen in Figures 4.25 and 4.26. The effect is dramatic but opposite in each algorithm. With Map Growing, as the degree increases so does the Average Transmitted Bits. This increase is more related to larger packets rather than to more packets. When a node localizes, it broadcasts its position estimate as well as a list of all the neighbors along with their estimated position if known. As the degree increases, each node knows about more

other nodes. This increases the size of each position broadcast packet, causing each node to transmit more as the degree rises. This is not the trend with AFL.

In AFL, the size of each transmission does not depend on the number of neighbors a node has, so it is fairly constant. As discussed earlier, if a node has a higher degree it has more information to refine with and is able to balance the position "forces" much quicker than if it had fewer nodes. As a result, the higher the degree, the less refinement iterations a node uses, the less messages and bits it needs to transmit. With lower degrees, more iterations are needed driving up transmissions. For 300 nodes, degree 16 networks required about 700 - 900 iterations, while degree 8 networks typically required over 2,000. After degree 12, increasing the degree has diminishing returns. The decrease in Average Transmitted Bits from degree 12 to degree 16 is much less than that between degree 8 and degree 12. This suggests that at or around this degree, the node is able to refine efficiently and a higher degree does not help as much. This is also supported by the ADE data.

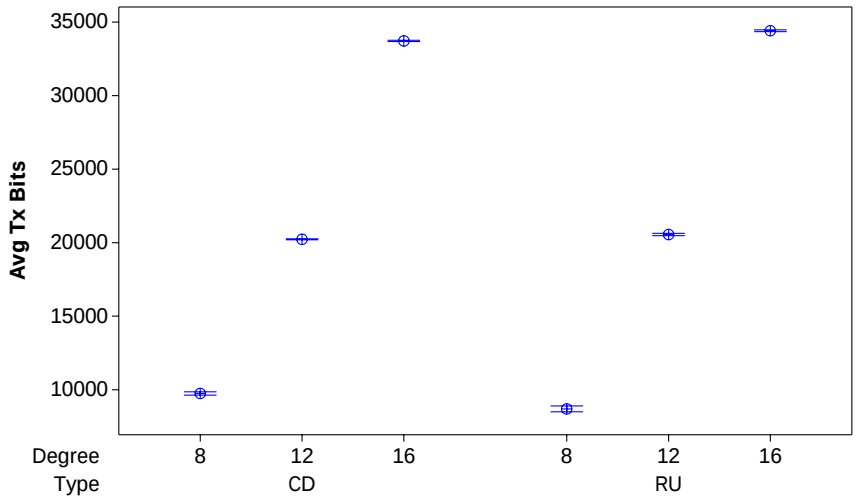


Figure 4.25: Map Growing Average Transmitted Bits vs Network Type and Degree

For Map Growing, the trend in Average Received Bits follows almost exactly the Average Transmitted Bits trend. As seen in Figure 4.27, with higher degrees, nodes broadcast with from about more neighbors increasing the message size and increasing the number of bits received at the same time. The trend for AFL Average

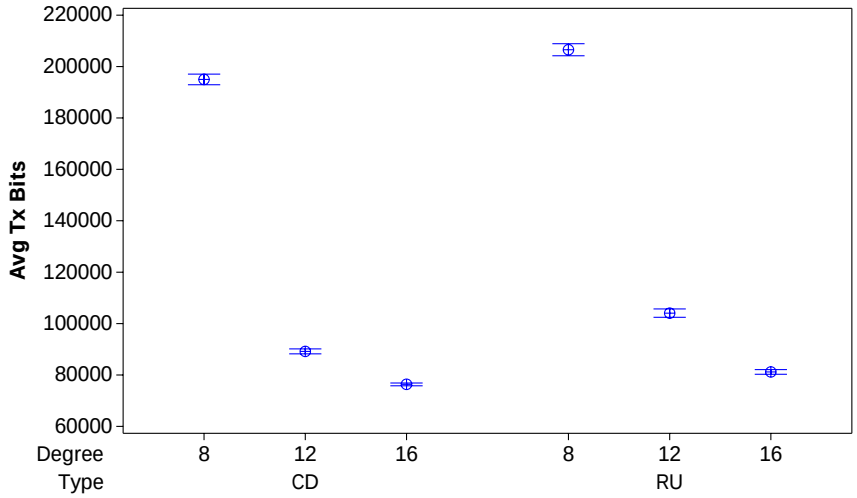


Figure 4.26: AFL Average Transmitted Bits vs Network Type and Range Error

Received Bits does not follow the Average Transmitted Bits as closely. As the degree increases from 8 to 12, the Average Received Bits drops in much the same way as it did for Average Transmitted Bits. However, between degree 12 and 16, the Average Received Bits actually rises significantly. This counterintuitive trend is actually the result of two separate phenomenon occurring at opposite ends of the factor levels. As the degree rises, the most influential trend is the one discussed earlier. The more neighbors a node has the more information it can use to refine to a balanced position quicker. Above degree 12, another trend is at work. For each message broadcast, a node's entire neighborhood hears it. So as the degree rises, the number of nodes that hear a single message also rises. Between degree 8 and 12, this effect is not as strong as the quicker refinement effect. Above degree 12, when the benefit of increasing degree diminishes, the effect of having more listeners becomes dominant. As a result the Average Received Bits is significantly higher for degree 16 networks. For degrees greater than 16, the trend should continue and received bits should rise more. This effect also explains why the Average Received Bits increases more sharply between degree 12 and 16 for Map Growing.

4.7.6 Computation of Effects on Communication. Figures 4.29 to 4.32 plot the main effects of each factor on Average Transmitted Bits and Average Received

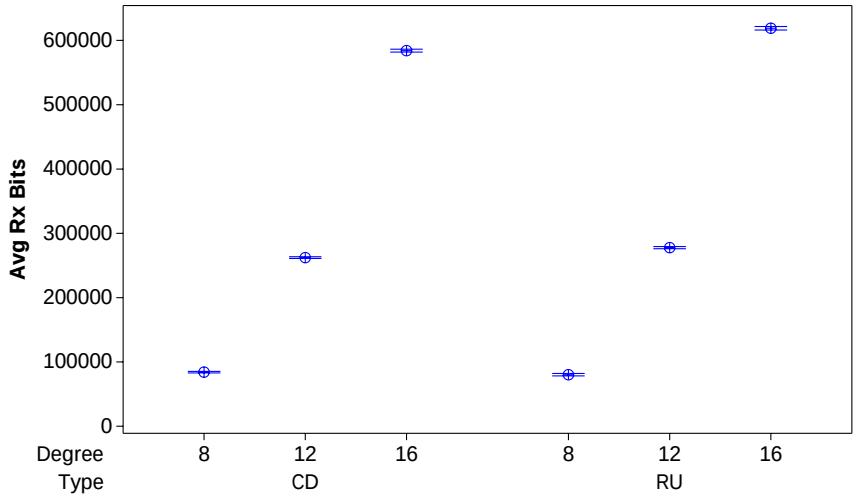


Figure 4.27: Map Growing Average Received Bits vs Network Type and Degree

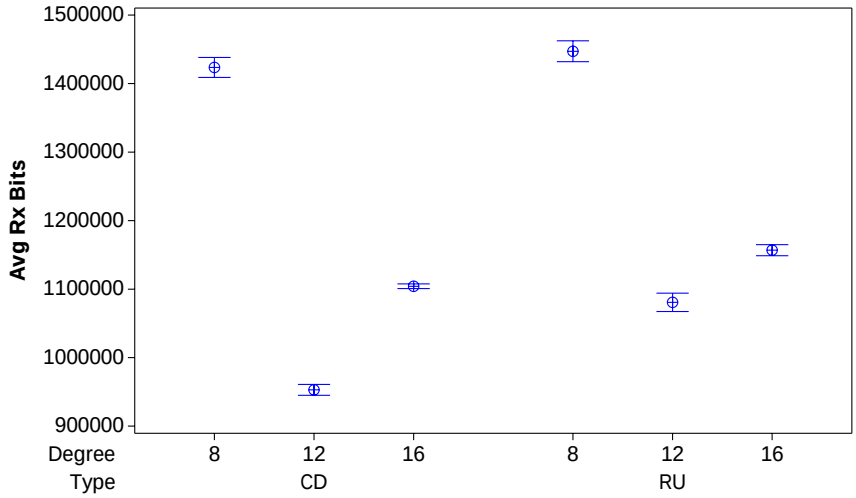


Figure 4.28: AFL Average Received Bits vs Network Type and Degree

Bits for both algorithms. In all four figures, range error is shown to have very little if any effect on either response for either algorithm. The network type and network size also does not effect Map Growing very much. While network type and network size do affect the communication of AFL, the factor that has the largest effects for both algorithms is degree. The degree drives Average Transmitted and Received Bits up for Map Growing. At the same time, higher degrees require fewer transmitted bits for AFL. The received bits reflects two phenomenon at work. First, the Average Received Bits drop due to quicker refinement times for higher degrees. Then, it rises

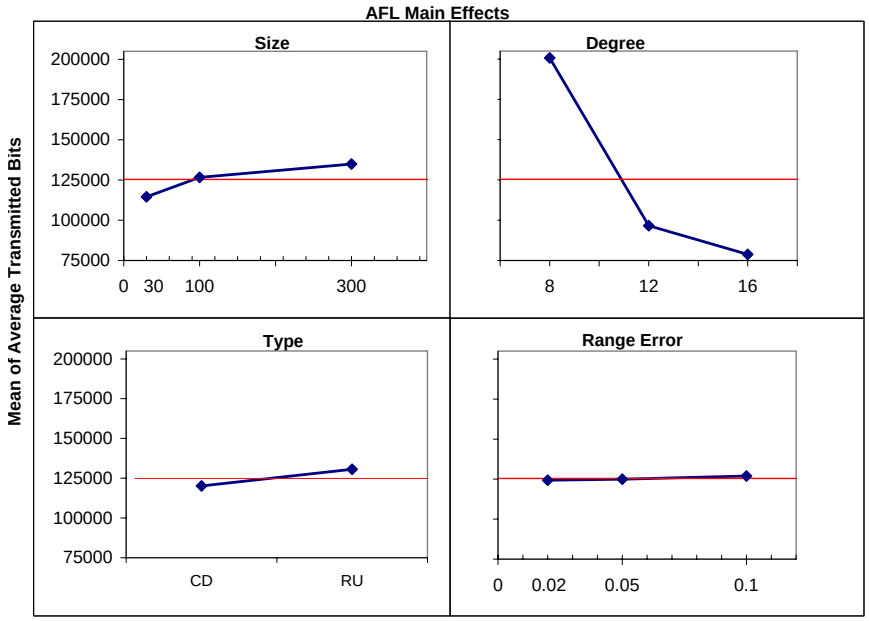


Figure 4.29: AFL Main Effects Plot of Average Transmitted Bits

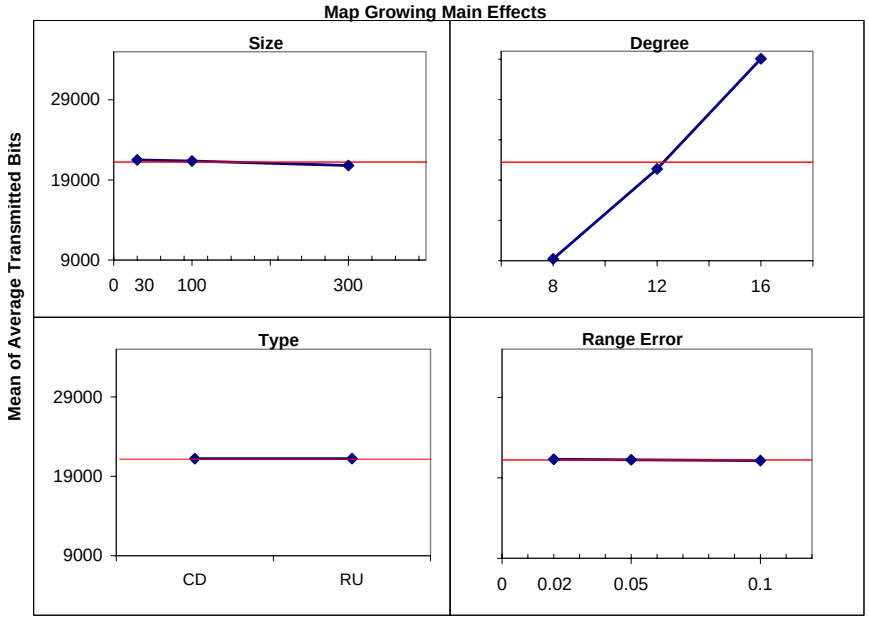


Figure 4.30: Map Growing Main Effects Plot of Average Transmitted Bits

for degree 16, since more nodes hear each broadcast and the speed of refinement has less of an effect. The tables for the effects of each factor on Average Transmitted and Received Bits is included in Tables H.2, H.5, H.14 and H.17 in Appendix H.

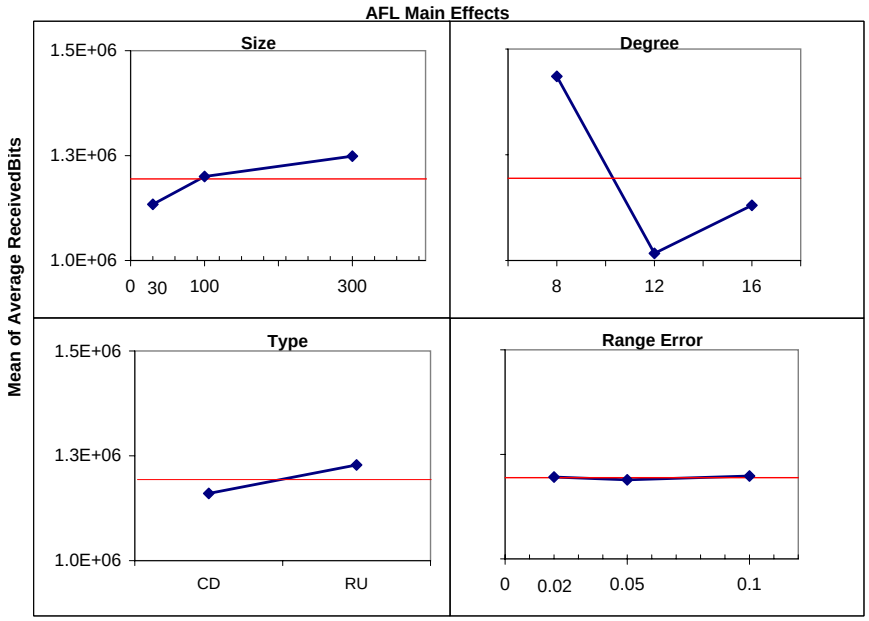


Figure 4.31: AFL Main Effects Plot of Average Received Bits

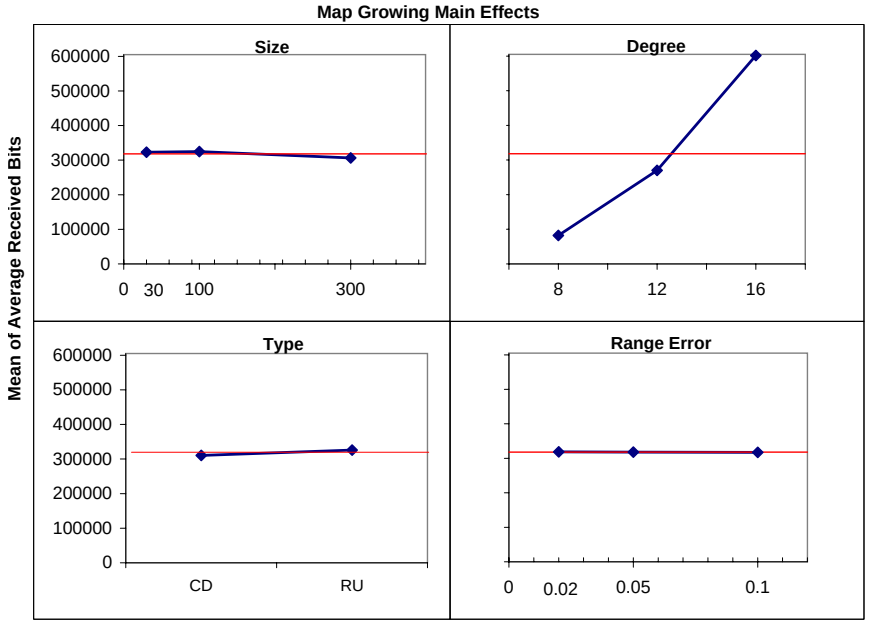


Figure 4.32: Map Growing Main Effects Plot of Average Received Bits

4.7.7 Energy ANOVA. To determine which factors account for variation in localization communication more than variation due to errors, an ANOVA is used. The ANOVA identifies how much variation in the system is due to each response and

how much is due to random error. For the results of an ANOVA to be valid, several assumptions must be met [Jai91]:

1. The effects of various factors are additive.
2. Errors are additive.
3. Errors are independent of the factor levels.
4. Errors are normally distributed.
5. Errors have the same variance for all factor levels.

Several visual tests are used to verify these assumptions. First a normal probability quantile-quantile plot of the residuals is prepared and checked to confirm that the plot is approximately linear. If so, normality of the errors can be assumed. A scatter plot of the residuals versus fitted values is checked to verify that there are no trends in the residuals. This makes sure the errors are independent. This plot is also checked to confirm there is no trend in the spread of the residuals, verifying that errors have the same variance for all factor levels. For each ANOVA presented, these plots are used and are included in Appendix F. Also included in Appendix F is a discussion regarding the reasoning behind the natural log transformation of the AFL data, the removal of some extreme outliers, and the partitioning of the Map Growing data. The AFL ANOVA uses the natural log of both Average Transmitted and Received Bits to adjust for a wide variance in response values and non-homoscedatic data (non-constant variance). The Map Growing ANOVA does not include degree 8 results, due to the different mode of operation when networks localize fewer than 75% of the nodes. Once an ANOVA is run, the p-value of each factor is examined. In this case, the p-value is the probability that the variation attributed to the factor is actually due to measurement errors. This is also a sign that the factor is not significant to the response. If the p-value is greater than the α (for 90% Confidence Intervals, $\alpha = 0.1$), then it is probable the variation attributed to that factor is actually due to measurement error and the factor is pooled. When a factor is pooled, it and all interaction terms that include it are removed from the ANOVA and its variation is

attributed to measurement errors. The ANOVAs presented are the final versions with insignificant factors not included. The communication ANOVAs for both AFL and Map Growing are shown in Tables 4.4 to 4.7.

Table 4.4: Map Growing Average Transmitted Bits ANOVA

Component		Sum of Squares	Percentage of Variation	DOF	Mean Square	F Computed	p
y	SSY	0		828			
$\bar{y}$	SS0	0		1			
$y - \bar{y}$	SST	3.87E+10	100.00%	827			
Size	SSA	15996552	0.04%	2	7998276	76.82	< 0.0005
Degree	SSB	3.85E+10	99.52%	1	3.85E+10	369472.2	< 0.0005
Type	SSC	59833423	0.15%	1	59833423	574.64	< 0.0005
Size*Degree	SSAB	4989188	0.01%	2	2494594	23.96	< 0.0005
Size*Type	SSAC	16785803	0.04%	2	8392902	80.61	< 0.0005
Degree*Type	SSBC	1993943	0.01%	1	1993943	19.15	< 0.0005
Size*Degree*Type	SSABC	964241	0.00%	2	482120.5	4.63	0.01
Error	SSE	84964461	0.22%	816	104123.1		

R-Sq(adj) = 99.78% $s_e = \sqrt{MSE} = \sqrt{104123.114} = 322.6811$

The Map Growing Average Transmitted Bits ANOVA in Table 4.4 does not include range error as it is insignificant to the response. Degree is responsible for over 99.52% of the variation in Average Transmitted Bits and all other factors explain less than 1%, with 0.22% of the variation explained by error. This is consistent with the interval and main effects data discussed earlier. This suggests the best way to control transmissions when using Map Growing is to tightly control the degree. The coefficient of determination (R^2 adjusted) is 99.78%. This is a higher better "goodness of fit" metric which accounts for pooled factors and interactions. The value is the percentage of total variation explained by the regression model. Thus, this ANOVA provides a very good fit of the data, meaning the regression explains nearly all the variation in the response.

As shown in Table 4.5, the average node degree is responsible for 98.4% in received bits which follows the same trend as Average Transmitted Bits. This is not surprising since received bits and transmitted bits are closely related for Map Growing. Like the transmitted bits ANOVA, Range Error is insignificant and is removed. All

Table 4.5: Map Growing Average Received Bits ANOVA

Component		Sum of Squares	Percentage of Variation	DOF	Mean Square	F Computed	P
y	SSY	0		828			
$\bar{y}$	SS0	0		1			
$y - \bar{y}$	SST	2.29E+13	100.00%	827			
Size	SSA	5.40E+10	0.24%	2	2.7E+10	174.46	< 0.0005
Degree	SSB	2.25E+13	98.40%	1	2.25E+13	145206.8	< 0.0005
Type	SSC	1.19E+11	0.52%	1	1.19E+11	767.45	< 0.0005
Size*Degree	SSAB	10747096585	0.05%	2	5.37E+09	34.7	< 0.0005
Size*Type	SSAC	39208863813	0.17%	2	1.96E+10	126.61	< 0.0005
Degree*Type	SSBC	11934033382	0.05%	1	1.19E+10	77.07	< 0.0005
Size*Degree*Type	SSABC	4671349523	0.02%	2	2.34E+09	15.08	< 0.0005
Error	SSE	1.26E+11	0.55%	816	1.55E+08		
R-Sq(adj) = 99.44%			$s_e = \sqrt{MSE} = \sqrt{154845588.2} = 12443.7$				

other factors account for less than 1% of the variation with 0.55% of the variation attributed to error. The coefficient of determination is computed as 99.44%, indicating a very good fit of the data.

The AFL ANOVA of the natural log of Average Transmitted Bits (ln(ATB)) is presented in Table 4.6. The pooled interactions are Type*Range Error and Size*Degree*Range Error and all others that include these interactions. From the table, it is clear that degree is the most important factor in this situation as well. Degree accounts for 92.57% of the variation in ln(ATB). Network size is the next significant factor, responsible for 3.31% of the variation. All other factors and interactions are less than 1% with 2.56% of the variation attributed to error. The goodness of fit for this ANOVA is 97.38%, again indicating the ANOVA describes the behavior of the data very well.

The AFL ANOVA on the natural log of Average Received Bits (ln(ARB)) is presented in Table 4.7. The pooled interactions are also Size*Type*Range Error and Degree*Type*Range Error and the four factor interaction. Degree is also the most important factor in this ANOVA, accounting for 75.94% of the variation. Size explains 6.34% of the variation, while Type explains 2.41%. Variation due to error accounts for

Table 4.6: AFL ln(Average Transmitted Bits) ANOVA

Component		Sum of Squares	Percentage of Variation	DOF	Mean Square	F Computed	p
y	SSY	153906		1134			
$\bar{y}$	SS0	153703.5		1			
$y - \bar{y}$	SST	202.4766	100.00%	1133			
Size	SSA	6.7018	3.31%	2	3.3509	737.96	< 0.0005
Degree	SSB	187.4295	92.57%	2	93.71475	19984.93	< 0.0005
Type	SSC	1.7426	0.86%	1	1.7426	372.28	< 0.0005
Range Error	SSD	0.0708	0.03%	2	0.0354	7.6	0.001
Size*Degree	SSAB	0.3604	0.18%	4	0.0901	19.2	< 0.0005
Size*Type	SSAC	0.2373	0.12%	2	0.11865	25.54	< 0.0005
Size*Range Error	SSAD	0.2243	0.11%	4	0.056075	11.89	< 0.0005
Degree*Type	SSBC	0.3298	0.16%	2	0.1649	35.31	< 0.0005
Degree*Range Error	SSBD	0.078	0.04%	4	0.0195	4.11	0.003
Size*Degree*Type	SSABC	0.118	0.06%	4	0.0295	6.29	< 0.0005
Error	SSE	5.184	2.56%	1106	0.004687		
R-Sq(adj) = 97.38%			$s_\epsilon = \sqrt{MSE} = \sqrt{0.0046872} = 0.068463$				

11.66%. The coefficient of determination is 88.03%. This indicates the ANOVA that does not fit the data as well as the previous. There is some variation in the ln(ARB) not explained by the regression. This is likely due to the behavior of Average Received Bits as degree increases. As discussed earlier, the Average Received Bits initially drops when degree is increased from 8 to 12, then rises when degree is increased to 16. The initial drop is due to the less iterations required to refine the position when nodes have a higher degree. After degree 12, this effect diminishes. At this point the fact that more nodes can hear each message is more relevant. This drives up received bits more than the shorter refinement drives it down. Since this behavior is not linear, it cannot be accounted for by an ANOVA that requires the response to be additive. While the coefficient of determination is not as high as desired, it is high enough to be useful as rough order magnitude estimate of the variation in the response.

4.8 Energy Consumption Model

To predict energy consumption in configurations not tested in this work, a model is required. As stated in Chapter II, the energy used by a node is the sum of the

Table 4.7: AFL ln(Average Received Bits) ANOVA

Component		Sum of Squares	Percentage of Variation	DOF	Mean Square	F Computed	p
y	SSY	221491.7		1134			
$\bar{y}$	SS0	221457.6		1			
$y - \bar{y}$	SST	34.1447	100.00%	1133			
Size	SSA	2.1293	6.24%	2	1.06465	299.64	< 0.0005
Degree	SSB	25.9281	75.94%	2	12.96405	3598.86	< 0.0005
Type	SSC	0.8244	2.41%	1	0.8244	227.37	< 0.0005
Range Error	SSD	0.0221	0.06%	2	0.01105	3.08	0.047
Size*Degree	SSAB	0.166	0.49%	4	0.0415	11.44	< 0.0005
Size*Type	SSAC	0.1177	0.34%	2	0.05885	16.56	< 0.0005
Size*Range Error	SSAD	0.2738	0.80%	4	0.06845	18.84	< 0.0005
Degree*Type	SSBC	0.3753	1.10%	2	0.18765	52.39	< 0.0005
Degree*Range Error	SSBD	0.0758	0.22%	4	0.01895	5.14	< 0.0005
Type*Range Error	SSCD	0.0424	0.12%	2	0.0212	5.72	0.003
Size*Degree*Type	SSABC	0.2088	0.61%	4	0.0522	14.48	0
Error	SSE	3.9809	11.66%	1104	0.003606		
R-Sq(adj) = 88.03%		$s_e = \sqrt{MSE} = \sqrt{0.0036059} = 0.060049$					

energy used from transmitting, receiving, sensing and processing. Sensing is not used in localization so the Energy Consumption Model is

$$Energy = E_{Transmission} + E_{Receives} + E_{Processing} \tag{4.5}$$

For each algorithm, a multiple linear regression predicts the number of bits transmitted and received by a node in the average case. The amount of processing executed by a node is not collected in any of the experiments. To approximate this portion of the model, a worst-case analysis of the code for each algorithm is performed to produce an upper limit of the number of lines of code executed by a node using that algorithm. Once the number of bits transmitted and received and the number of processing cycles are known, each is converted to energy consumed, using conversions from typical wireless sensor nodes. These values are combined to form the total energy used to localize the network. The total from this method assumes that the receiver is turned on and off precisely when bits start to arrive and when they are done arriving. In

reality, the node receiver will be on for more time than just when it is receiving bits. To account for this, an alternate version of the model is presented that assumes the receiver is on continuously during localization. The time to localize is taken from a worst case estimate for both algorithms based on the size of the network.

The multiple linear regression for bits transmitted and received, or the natural log of bits transmitted and received, is based on a model of the factors and errors in the system. Since the network type factor is categorical and not numerical, CD is assigned the value of 1 and RU is assigned 0. The model is

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + ... + b_kx_k + e \tag{4.6}$$

where b_i are the different coefficients, x_i are the different factors, k is the number of predictors, and e is the error. The standard deviation of the prediction ($s_{\hat{y}_p}$), the standard deviation of the parameters (s_{b_j}) and the confidence interval on the mean of m future observations ($\hat{y}_p$) are

$$s_{\hat{y}_p} = s_e\sqrt{\left\{\frac{1}{m} + \mathbf{x}_p^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_p\right\}} \tag{4.7}$$

$$s_{b_j} = s_e\sqrt{C_{jj}} \tag{4.8}$$

$$\text{Lower CI} = \hat{y}_p - s_{\hat{y}_p}t_{[1-\alpha/2;n-k-1]} \tag{4.9}$$

$$\text{Upper CI} = \hat{y}_p + s_{\hat{y}_p}t_{[1-\alpha/2;n-k-1]} \tag{4.10}$$

where n is the number of samples [Jai91]. The vector $\mathbf{x}_p$ is $(1, x_{1p}, x_{2p}, ..., x_{kp})$ which are the inputs to the predictor variables. $\mathbf{X}$ is a matrix of sample inputs taken from the experiment. The matrix $\mathbf{x}_p^T(\mathbf{X}^T\mathbf{X})^{-1}$ is also referred to as C and is included in Appendix H for all regressions. Since n is high (828 for Map Growing, 1134 for AFL), the z-variate is used at $\alpha = 0.1$, for 90% confidence intervals (1.645). To determine the best regression, different combinations of first, second and third order terms were examined. The R^2(adj) value is checked. The highest combinations are kept. If a lower order regression is within 1% of a higher order regression with a better R^2(adj),

the simpler, low order regression is used. For Map Growing, the regressions are only valid for networks with degree higher than 8, since degree 8 networks were removed as a separate mode of operation. The AFL regressions are on ln(ATB) and ln(ARB) and are converted as necessary for the Energy Consumption Model. The symbols in Table 4.8 are used in the development of the regressions.

Table 4.8: Regression Symbols

Symbol	**Meaning**
S	Network Size
D	Average Degree of a Network
T	Network Type (CD = 1, RU = 0)

4.8.1 Map Growing Regressions. The Map Growing regression for Average Transmitted Bits (ATB) is listed in (4.11) with the corresponding coefficient of determination and standard deviation of errors. Note the regression does not include range error, and that coefficient for the degree parameter is much greater than the coefficient for size and type.

$$\text{ATB} = -20123 - 0.574S + 3408D - 538T \tag{4.11}$$

$$R^2(adj) = 99.7 \tag{4.12}$$

$$s_e = 384.986 \tag{4.13}$$

From the above information, the regression data in Table 4.9 is generated. With a high coefficient of determination, this regression explains a high percentage of the variation and closely fits the data. Using various example configurations, the regression in

Table 4.9: Map Growing Regression Data for Average Transmitted Bits

Predictor	b_i	s_{b_j}	**Lower CL**	**Upper CL**	**p**
Constant	-20123.3	97	-20282.865	-19963.7	< 0.0005
Size	-0.5736	0.1169	-0.7659005	-0.3813	< 0.0005
Degree	3408.15	6.69	3397.14495	3419.155	< 0.0005
Type	-537.63	26.76	-581.6502	-493.61	< 0.0005

Equation 4.11 is evaluated for 1, 10 and 100 future observations (m). The predicted

ATB, standard deviation ($s_{\hat{y}_p}$) and 90% confidence intervals on the prediction are provided in Table 4.10.

Table 4.10: Map Growing Example Regression Results, Average Transmitted Bits

S	D	T	ATB	m	$s_{\hat{y}_p}$	Lower CL	Upper CL
400	14	CD	26823.73	1	386.857	26187.35	27460.11
400	14	CD	26823.73	10	127.536	26613.93	27033.53
400	14	CD	26823.73	100	54.095	26734.74	26912.72
200	10	RU	13843.48	1	386.495	13207.7	14479.26
200	10	RU	13843.48	10	126.436	13635.49	14051.47
200	10	RU	13843.48	100	51.447	13758.85	13928.11
1000	18	CD	39453.19	1	400.594	39453.19	40771.15
1000	18	CD	39453.19	10	164.569	39841.45	40382.89
1000	18	CD	39453.19	100	117.234	39919.32	40305.02

The Map Growing regression for Average Received Bits (ARB) (4.14) along with R^2(adj) and standard deviation of errors is shown below. The high R^2(adj) indicates a regression that explains nearly all of the variation. Again, the coefficient for the degree parameter is by far the largest and range error is not included.

$$\text{ARB} = -697537 - 62.7S + 82394D - 23960T \tag{4.14}$$

$$R^2 = 99.1 \tag{4.15}$$

$$s_e = 15750.0 \tag{4.16}$$

Table 4.11 lists the standard deviation of each parameter as well as the upper and lower confidence levels and p-value for each parameter. Using the same example con-

Table 4.11: Map Growing Regression Data for Average Received Bits

Predictor	Mean	St Dev	Lower CL	Upper CL	p
Constant	-697537	3969	-704066.005	-691007.995	< 0.0005
Size	-62.652	4.784	-70.52168	-54.78232	< 0.0005
Degree	82394.3	273.7	81944.0635	82844.5365	< 0.0005
Type	-23960	1095	-25761.275	-22158.725	< 0.0005

figurations, the regression in (4.14) is evaluated for 1, 10 and 100 future observations

(m). The predicted ATB, standard deviation, and 90% confidence intervals on the prediction are provided in Table 4.12.

Table 4.12: Example Map Growing Regression Results for Average Received Bits

S	D	T	ARB	m	$s_{\hat{y}_p}$	Lower CL	Upper CL
400	14	CD	406962.4	1	15826.51	380927.8	432997.01
400	14	CD	406962.4	10	5217.58	17165.83	415545.32
400	14	CD	406962.4	100	2213.05	403321.9	410602.86
200	10	RU	113875.6	1	15811.73	87865.31	139885.9
200	10	RU	113875.6	10	5172.56	105366.7	122384.46
200	10	RU	113875.6	100	2104.72	110413.3	117337.87
1000	18	CD	698948.4	1	16388.52	671989.3	725907.52
1000	18	CD	698948.4	10	6732.62	687873.2	710023.57
1000	18	CD	698948.4	100	4796.11	691058.8	706838.01

4.8.2 AFL Regressions. The AFL regression on ln(ATB) is presented in (4.17). Like the previous two regressions, range error is not included and there is a strong fit of the data with a high R^2(adj) value of 96.4%. Again degree is the most significant factor, but in this case the coefficient is negative. This reflects how the transmitted bits tend to drop as degree increases. The conversion to ATB is included in (4.20).

$$ln(\text{ATB}) = 15.4 + 0.000647S - 0.547D - 0.0786T + 0.0179D^2 \quad (4.17)$$

$$R^2 = 96.4 \quad (4.18)$$

$$s_e = 0.0804320 \quad (4.19)$$

$$\text{ATB} = e^{15.4+0.000647S-0.547D-0.0786T+0.0179D^2} \quad (4.20)$$

The standard deviations of the parameters, their confidence intervals and p-values are listed in Table 4.13.

Table 4.14 shows the results from running the regression for ln(ATB) on the three example configurations. In addition to the prediction, standard deviation and confidence interval, the data is converted from ln(bits) to bits.

Table 4.13: AFL Regression Data for ln(ATB)

Predictor	**Mean**	**St Dev**	**Lower CL**	**Upper CL**	**p**
Constant	15.3847	0.0434	15.313307	15.456	< 0.0005
Size	0.00064658	0.00002086	0.00061227	0.0007	< 0.0005
Degree	-0.547101	0.007635	-0.5596606	-0.5345	< 0.0005
Type	-0.078613	0.004777	-0.0864712	-0.0708	< 0.0005
Degree^2	0.0179057	0.0003167	0.0179055	0.0179	< 0.0005

Table 4.14: AFL Example Regression Results for ln(ATB) with transformation to ATB

S	**D**	**T**	**ln(ATB)**	**m**	$s_{\hat{y}_p}$	**Lower CL**	**Upper CL**	**ATB**	**Lower CL**	**Upper CL**
400	14	CD	11.4148	1	0.07794	11.2866	11.543	90653.52	79745.85	103053.1
400	14	CD	11.4148	10	0.02598	11.3721	11.4576	90653.52	86864.09	94617.72
400	14	CD	11.4148	100	0.01163	11.3957	11.4339	90653.52	88938.46	92401.64
200	10	RU	11.8336	1	0.07763	11.7059	11.9613	137805.7	121285.2	156576.5
200	10	RU	11.8336	10	0.02502	11.7924	11.8747	137805.7	132243.5	143587.5
200	10	RU	11.8336	100	0.009276	11.8183	11.8488	137805.7	135713.3	139916.3
1000	18	CD	11.9063	1	0.08112	11.7728	12.0397	148197.3	129676.7	169346.1
1000	18	CD	11.9063	10	0.03436	11.8498	11.9628	148197.3	140056.3	156811.5
1000	18	CD	11.9063	100	0.02532	11.8646	11.9479	148197.3	142144.6	154492.4

The AFL regression on ln(ARB) is (4.21) below. Also listed is the standard deviation of errors and coefficient of determination of 84.5%. A low value is expected since the corresponding ANOVA also has a lower than desired R^2(adj) value. This indicates the regression does not explain all of the variation in ln(ARB). The unexplained variation is due to the same factor discussed with the ANOVA for AFL ln(ARB). The factor causing the unexplained variation is the two opposite trends in the Average Received Bits (high degree nodes use fewer iterations requiring fewer messages to refine, higher degree nodes also cause more nodes to hear each message). While the R^2(adj) is lower than desired, it is acceptable for rough order of magnitude estimation on the natural log of average received bits. The conversion to Average Received Bits (ARB) is included in (4.24).

$$ln(\text{ARB}) = 16.3 + 0.00108S - 0.397D - 0.0539T - 2\text{E-}06S^2 + 0.0153D^2 \quad (4.21)$$

$$R^2 = 84.5 \quad (4.22)$$

$$s_e = 0.0683044 \quad (4.23)$$

$$\text{ARB} = e^{16.3+0.00108S-0.397D-0.0539T-2\text{E-}06S^2+0.0153D^2} \quad (4.24)$$

Table 4.13 lists the predictors, their coefficients as well as their standard deviations, confidence intervals and p-values. The results of the regression (4.21) using the three

Table 4.15: AFL Regression Data for ln(ARB)

Predictor	**Mean**	**St Dev**	**Lower CL**	**Upper CL**	**p**
Constant	16.3148	0.0371	16.254	16.376	< 0.0005
Size	0.0010809	0.0001116	0.0009	0.0013	< 0.0005
Degree	-0.396969	0.006484	-0.4076	-0.3863	< 0.0005
Type	-0.053926	0.004057	-0.0606	-0.0473	< 0.0005
Size^2	-2.08E-06	0.00000032	-3E-06	-2E-06	< 0.0005
Degree^2	0.015293	0.0002689	0.01485066	0.015735341	< 0.0005

example configurations is shown in Table 4.16, including conversions from ln(bits) to bits.

Table 4.16: AFL Example Regression Results for ln(ARB) with transformation to ARB

S	**D**	**T**	**ln(ARB)**	**m**	$s_{\hat{y}_p}$	**Lower CL**	**Upper CL**	**ARB**	**Lower CL**	**Upper CL**
400	14	CD	13.8003	1	0.08035	13.6681	13.9324	984904.5	862939.6	1123995.0
400	14	CD	13.8003	10	0.04751	13.7221	13.8785	984904.5	910819.5	1065016.0
400	14	CD	13.8003	100	0.04287	13.7298	13.8708	984904.5	917859.9	1056846.0
200	10	RU	14.0074	1	0.07738	13.8908	14.124	1211537.0	1078196.0	1361367.0
200	10	RU	14.0074	10	0.02877	13.9601	14.0547	1211537.0	1155565.0	1270219.0
200	10	RU	14.0074	100	0.02019	13.9742	14.0406	1211537.0	1171974.0	1252435.0
1000	18	CD	13.07126	1	0.1323	12.8536	13.2889	475092.1	382162.0	590603.4
1000	18	CD	13.07126	10	0.11537	12.8815	13.261	475092.1	392974.5	574353.3
1000	18	CD	13.07126	100	0.1135	12.8844	13.258	475092.1	394115.8	572632.8

4.8.3 Processor Usage. To estimate the power used due to processing, a worst case analysis is done on both algorithms to estimate the upper bound on the number of instructions processed by a node to localize the network. An upper bound ensures that the true power due to processing does not exceed the estimate. The analysis is executed by altering the localization OPNET C code so it can be compiled on an Intel x86 CPU. OPNET specific library calls and print statements and other functions not available on a WSN are either removed or replaced with analogous code. After compilation, the assembly code is examined. The number of assembly lines of code are counted for each state and function. Then the number of times each state and function are executed is estimated as function of the highest degree node found in

the experiment. The highest degree node in all 540 networks had 36 neighbors. The numbers of assembly lines of code executed for each state and function are summed for a total representing the worst case estimate for number of instructions executed to localize with that particular algorithm. It is assumed that each instruction requires roughly 1 cycle to execute. Table 4.17 show the results of the analysis.

Table 4.17: Worst Case Estimate for Instructions Executed during Localization

Algorithm	**Instructions**
Map Growing	134,318,931
AFL	2,295,169

Map Growing is by far more processor intensive. This is largely due to its localization process that attempts to 3 Beacon localize or 2 Beacon localize with all combinations of 3 or 2 neighbors in its neighbor list. Attempting all possible combinations available allows it to achieve the result with the lowest residual. This leads to double or triple "for" loops through all neighbors, which quickly drives up the number of instructions executed. Actual implementations may limit the number of combinations attempted or optimize the search for the best. AFL is simpler mainly because it does little processing through a list of neighbors. During the reference node search phase, each node does need to keep track of the lowest hop count to each reference node. This can require a single pass search through its list of neighbors for each neighbor. In AFL's second phase, everything also completed in a single pass. For most messages received, the information is only used to update the position of a neighbor. For the remaining, the node actually refines its position but only requires a single pass through the neighbor list to accomplish this. Even with the potential for very large numbers of messages transmitted and received, this simplicity keeps the number of instructions executed relatively low.

4.8.4 Power Conversions. With the number of bits transmitted and received and instructions processed by a node, each can be converted to an amount of energy consumed by the node. Since batteries are typically rated in mA-hr, these are the units

used to determine how much of the battery is actually used by a node using a particular localization algorithm and configuration. The General Energy Consumption Model (adapted from [ASSC02]) in (4.29) produces the number of mA-hr used given the number of bits transmitted and received, and instructions processed, and the relevant node parameters. Table 4.18 lists all the symbols used in the conversions and the development of the model.

Table 4.18: Table of Symbols

Symbol	**Meaning**
b_T	Bits Transmitted
b_R	Bits Received
I	Instructions Executed by an Algorithm
r	Data Rate (bits/sec)
h	Processor Clock Rate (hertz)
d_T	Current Draw from Transmitting (mA)
d_R	Current Draw from Receiving (mA)
d_P	Current Draw from Processing (mA)
E_{Tx}	Energy Consumed from Transmitting (mA-hr)
E_{Rx}	Energy Consumed from Receiving (mA-hr)
E_P	Energy Consumed from Processing(mA-hr)
E	Total Energy Consumed (mA-hr)

$$E_{Tx} = \frac{b_T \times d_T}{3600r} \tag{4.25}$$

$$E_{Rx} = \frac{b_R \times d_R}{3600r} \tag{4.26}$$

$$E_P = \frac{I \times d_P}{3600h} \tag{4.27}$$

$$E = E_{Tx} + E_{Rx} + E_P \tag{4.28}$$

The General Energy Consumption Model is

$$E = \frac{b_T \times d_T}{\left(3600\frac{\text{s}}{\text{hr}}\right) r} + \frac{b_R \times d_R}{\left(3600\frac{\text{s}}{\text{hr}}\right) r} + \frac{I \times d_P}{\left(3600\frac{\text{s}}{\text{hr}}\right) h}. \tag{4.29}$$

4.8.5 Energy Consumption Model. Using the Map Growing regressions for ATB, ARB ((4.11) and (4.14)) in place of b_T and b_R and the upper bound estimate for instructions executed in place of I, the complete General Map Growing Energy Consumption Model is shown in (4.31). This model predicts the upper bound of mA-hr used by an average node to localize using Map Growing, given the size, degree and type of network. (Note: The usefulness of this model degrades for degrees 8 or less because of the different communication behavior of Map Growing when it can not localize over 90% of the connected nodes.)

The General Map Growing Energy Consumption Model is

$$\begin{aligned} E &= (-20123 - 0.574S + 3408D - 538T) \times \frac{d_T}{\left(3600\frac{\text{s}}{\text{hr}}\right) r} \\ &\quad +(-697537 - 62.7S + 82394D - 23960T) \times \frac{d_R}{\left(3600\frac{\text{s}}{\text{hr}}\right) r} \\ &\quad +134,318,931\text{instr} \times \frac{d_P}{\left(3600\frac{\text{s}}{\text{hr}}\right) h}. \end{aligned} \tag{4.30}$$

Actual values from wireless nodes can be used for r, h, d_T, d_R, and d_P. The MICA2 Wireless Measurement System is a wireless node built and distributed by Crossbow Technology Inc [Cro05]. This node is commonly used in current wireless sensor network research [AV04]. The summary of MICA2 data is presented in Table 4.19. Using

Table 4.19: MICA2 Data Summary [Cro05]

Processor Full Operation	8mA (7.37Mhz)
Radio, Receive	10mA
Radio, Transmit (full power)	27mA
Data Rate	38.4 Kbaud
Typical Battery	2×AA (Alkaline)
Typical Battery Capacity	2000mA-hr
Operating Voltage Range	3.6 to 2.7 Volts

these values, the Map Growing Energy Consumption Model for the MICA2 is

$$\begin{aligned} E &= \frac{(-20123 - 0.574S + 3408D - 538T) \times 27\text{mA}}{\left(3600\frac{\text{s}}{\text{hr}}\right)\left(38400\frac{\text{bits}}{\text{s}}\right)} \\ &\quad + \frac{(-697537 - 62.7S + 82394D - 23960T) \times 10\text{mA}}{\left(3600\frac{\text{s}}{\text{hr}}\right)\left(38400\frac{\text{bits}}{\text{s}}\right)} \\ &\quad + \frac{134,318,931 \times 8\text{mA}}{\left(3600\frac{\text{s}}{\text{hr}}\right)7.37 \times 10^6\frac{\text{instr}}{\text{s}}} \end{aligned} \tag{4.31}$$

$$\begin{aligned} E &= (-20123 - 0.574S + 3408D - 538T) \times 1.95313\text{E-}07\frac{\text{mA-hr}}{\text{bit}} \\ &\quad + (-697537 - 62.7S + 82394D - 23960T) \times 7.2338\text{E-}08\frac{\text{mA-hr}}{\text{bit}} \\ &\quad + 0.0405002\text{mA-hr} \end{aligned} \tag{4.32}$$

The sample configurations presented earlier are again presented in Table 4.20 using the complete Map Growing Energy Consumption Model to estimate the mA-hr and percent of battery used. The data from the MICA2 wireless sensor node is used.

Table 4.20: Example Map Growing Energy Consumption Model Results

S	D	T	Predicted ATB	Predicted ARB	mA-hr Consumed	Batt.	% of Consumed
400	14	CD	26823.73	406962.4	0.075178		0.003759
200	10	RU	13843.48	113875.6	0.051442		0.002572
1000	18	CD	39453.19	698948.4	0.098766		0.004938
Average Map Growing Response			21225.04	317935.8	0.067645		0.003382

The AFL regressions for ln(ATB) and ln(ARB) ((4.17) and (4.21)) are transformed and then substituted into (4.29) along with the AFL upper bound instruction estimate to produce the complete General AFL Energy Consumption Model, shown in (4.34). This model predicts the upper bound of mA-hr used by an average node to localize using AFL, given the size, degree and type of network. Substituting the MICA2 values in provides the AFL Energy Consumption Model for the MICA2.

The General AFL Energy Consumption Model is

$$\begin{aligned} E &= e^{(15.4+0.000647S-0.547D-0.0786T+0.0179D^2)} \times \frac{d_T}{\left(3600\frac{\text{s}}{\text{hr}}\right) r} \\ &\quad + e^{(16.3+0.00108S-0.397D-0.0539T-2\text{E-}06S^2+0.0153D^2)} \times \frac{d_R}{\left(3600\frac{\text{s}}{\text{hr}}\right) r} \\ &\quad + 2,295,169\text{instr} \times \frac{d_P}{\left(3600\frac{\text{s}}{\text{hr}}\right) h}. \end{aligned} \tag{4.33}$$

The AFL Energy Consumption Model for the MICA2 is

$$\begin{aligned} E &= e^{(15.4+0.000647S-0.547D-0.0786T+0.0179D^2)} \times 1.95313\text{E-}07\frac{\text{mA-hr}}{\text{bit}} \\ &\quad + e^{(16.3+0.00108S-0.397D-0.0539T-2\text{E-}06S^2+0.0153D^2)} \times 7.2338\text{E-}08\frac{\text{mA-hr}}{\text{bit}} \\ &\quad + 0.000692046\text{mA-hr}. \end{aligned} \tag{4.34}$$

The sample configurations presented earlier are used with the AFL Energy Consumption Model for the MICA2 to estimate the mA-hr and percentage battery consumed. The results are shown in Table 4.21

Table 4.21: Example AFL Energy Consumption Model Results

S	D	T	Predicted ATB	Predicted ARB	mA-hr Consumed	% of Batt. Consumed
400	14	CD	90653.52	984904.54	0.08964	0.004482
200	10	RU	137805.70	1211536.57	0.1153	0.005762
1000	18	CD	148197.33	475092.12	0.0640	0.003200
Average AFL Response			125394.72	1194295.65	0.1116	0.005579

4.9 Worst Case Energy Consumption Model

The previous Energy Consumption Model assumes the receiver is turned on and off precisely when bits start to arrive and when they stop arriving. Actual receivers of wireless sensor nodes are controlled by MAC protocols that will likely obey sleep schedules to conserve power. Even in these cases, the receiver is probably on more than just the time it is receiving. To account for this time, the Worst Case

Energy Consumption Model is the same as the previous but assumes the receiver is on during the entire localization process. The time to localize is taken from the worst case estimate for both algorithms and is included in Table 4.22. This estimate is calculated by examining the number of messages and the amount of processing required as well as the size and degree of the network. The equations represent an upper bound estimate in the growth of running time as size increases. Running time for AFL depends on degree also, but small degrees are assumed to maintain an upper bound estimate. Similarly for Map Growing, large degrees are assumed. With these

Table 4.22: Approximate Running Times (min)

AFL	**Map Growing**
12+0.04Size	-0.3+0.041Size

equations the Worst Case General Energy Consumption Model is

$$E = \frac{b_T \times d_T}{\left(3600\frac{\text{s}}{\text{hr}}\right) r} + \frac{t_R \times d_R}{\left(3600\frac{\text{s}}{\text{hr}}\right)} + \frac{I \times d_P}{\left(3600\frac{\text{s}}{\text{hr}}\right) h}. \tag{4.35}$$

given that t_R is the time the receiver is on, which is defined by the equations in Table 4.22. From (4.35), the Worst Case Map Growing Energy Consumption Model for the MICA2 is

$$\begin{aligned} E &= (-20123 - 0.574S + 3408D - 538T) \times 1.95313\text{E-}07\frac{\text{mA-hr}}{\text{bit}} \\ &\quad + \frac{(-0.3 + 0.041S)\text{min} \times 60 \times \frac{\text{s}}{\text{min}} \times 10\text{mA}}{\left(3600\frac{\text{s}}{\text{hr}}\right)} \\ &\quad + 0.0405002\text{mA-hr} \\ E &= (-20123 - 0.574S + 3408D - 538T) \times 1.95313\text{E-}07\frac{\text{mA-hr}}{\text{bit}} \\ &\quad + (-0.05 + 6.8333\text{E-}3S)\text{mA-hr} \\ &\quad + 0.0405002\text{mA-hr}. \end{aligned} \tag{4.36}$$

The Worst Case AFL Energy Consumption Model for the MICA2 is

$$\begin{aligned} E &= e^{(15.4+0.000647S-0.547D-0.0786T+0.0179D^2)} \times 1.95313\text{E-}07\frac{\text{mA-hr}}{\text{bit}} \\ &\quad +(2+6.6667\text{E-}3S)\text{mA-hr} \\ &\quad +0.000692046\text{mA-hr}. \end{aligned} \tag{4.37}$$

Using the equations (4.37) and (4.38) and the example configurations presented previously, Tables 4.23 and 4.24 present the results for the Worst Case Energy Model for both Map Growing and AFL using the MICA2.

Table 4.23: Example Worst Case Map Growing Energy Consumption Model Results

Size	Degree	Type	Predicted ATB	Approx. Run Time(min)	mA-hr Consumed	% of Batt. Consumed
400	14	CD	26823.73	16.1	2.729072548	0.136453627
200	10	RU	13843.48	7.9	1.359870677	0.067993534
1000	18	CD	39453.19	40.7	6.831539240	0.341576962
Average Map Growing Response			21225.04	5.563	0.971812388	0.048590619

Table 4.24: Example Worst Case AFL Energy Consumption Model Results

Size	Degree	Type	Predicted ATB	Approx. Run Time(min)	mA-hr Consumed	% of Batt. Consumed
400	14	CD	90653.51582	28	4.685064477	0.234253224
200	10	RU	137805.6999	20	3.360940555	0.168047028
1000	18	CD	148197.3336	52	8.696303504	0.434815175
Average AFL Response			125394.72	17.72	2.978516535	0.148925827

4.10 Energy Consumption Findings

The development and application of the Energy Consumption Model for both AFL and Map Growing reveal that relatively very little energy is expended during localization. Among the example configurations, none expend more than a half of a percent of the battery. In addition to the MICA2, the model is applied to the same configurations using the MICA2DOT. The parameters and calculations are found in Appendix I. The only difference in this wireless node is the form factor is much smaller, the processor is less powerful (4MHz) and the battery capacity is much smaller (560mA-hr). Even on this smaller node and using the worst case model, none of the

example configurations expend more than 1.55% percent of the battery. The single most energy expensive simulation (AFL, 30 Nodes, degree 8, RU, RE = 0.02) uses at most 0.11% of the MICA2 battery and 0.4% of the MICA2DOT battery, under the worst case model.

Since the Energy Consumption Model is based on limited data and may not account for all behavior, it is only able to provide a rough order magnitude estimate of the energy costs due to localization. Simulating with more levels of each factor would provide more data and allow for more confidence in the model results. However, given these limitations, the model still reports small amounts of energy consumed in all cases. This is still true even when the receiver is assumed to be continuously on.

Even though, on average, AFL requires almost 6 times more transmitted bits and almost 4 times more received bits than Map Growing, neither use a significant amount of the battery. This result is contrary to much of the current localization research which aims to conserve overhead messages sent in an effort to conserve precious battery power [IS03] [NN01] [SPS02] [TP03]. This result implies less energy efficient localization algorithms can be tolerated, especially if they provide much better results in terms of accuracy, as in the case of AFL over Map Growing. Additionally, when energy does indeed need to be conserved, there is not much savings available within the localization algorithm. More savings will be found in the operation of the WSN, MAC protocol, and routing method.

Another goal of this work is to find the factors that most influence localization energy costs. According to the analysis on the communication computation of effects and ANOVA for both algorithms, node degree is by far the biggest factor (among Size, Type, Degree and Range Error) affecting localization communication costs. Node degree consistently had the largest effect, and explained the largest amount of variation in communication. No other factor is close. This means that if energy truly does need to be conserved down to the hundredth of a percent of battery capacity in the localization process the factor that should be tightly controlled is degree. Additionally,

the analysis illustrates the choice of an algorithm can have an relatively large effect on communication. AFL, the concurrent algorithm, required much more communication than Map Growing, the incremental algorithm. This trend is expected, but the relative size of the difference between the two algorithms is startling. AFL consistently requires orders of magnitude more communication that Map Growing. Despite the difference in communication performance, AFL outperforms Map Growing in terms of accuracy in every configuration, also by large amounts. Depending on the localization needs of a network, concurrent algorithms with refinement, like AFL, should be chosen if position accuracy is most important, while incremental algorithms, like Map Growing, should be chosen if coarse accuracy is sufficient but all sources of energy consumption need to be tightly controlled. Even though the experiment methodology assumes all messages are received, collisions can be tolerated in an actual system with the same conclusions.

4.11 Summary

The results of the WSN localization experiments show that algorithm type and degree are the largest factors affecting performance. AFL uses much more communication, while providing consistently much better position estimates. Also, increased degree drives Map Growing communication costs up, while it drives AFL communication costs down for degrees at or below 12. Beyond that, having many nodes hearing each message drives the number of received bits up. This dual effect on AFL received bits caused the ANOVA and regression for that response to have a low coefficient of determination. All others were very high. Regressions are used as part of an Energy Consumption Model to predict the energy used by a localization algorithm given parameters about the node and network conditions. Results from this model show that little energy is actually used during the localization process, reducing the need to conserve messages and the risk of partitioning the network.

V. Conclusions

5.1 Introduction

This chapter summarizes the objectives and outcomes of this research. First the objectives are reviewed and discussed to determine they are met. Next the impact of this research is presented, followed by an overview of the contributions of this research. Possible areas of future work are described last.

5.2 Meeting the Objectives and Impact

The primary goal of the research is to determine which factor is most important to energy consumption in wireless sensor network localization. The second goal is to identify which algorithm provides more accurate position estimates and is more energy efficient. The last goal is to develop an Energy Consumption Model that can be used to estimate the energy consumption without simulation or deployment of an actual system.

5.2.1 Energy Consumption Factor. For both AFL and Map Growing, node degree is the most important factor. Degree accounts for over 90% of the variation in Average Transmitted and Received Bits ANOVAs for Map Growing and over 90% and 75% percent in the respective ANOVAs for AFL. The only difference is that increased degree causes Map Growing communication to drastically increase, while it causes AFL transmissions to decrease. With more neighbors, Map Growing nodes send larger packets containing more information. AFL's refinement phase, which is always its most costly in terms of energy, is much shorter when nodes have larger degrees. Above degree 12, the number of receive bits rises due to the increased number of nodes that hear each message. The remaining factors have relatively very little effect. Since degree is by far most important to localization energy consumption, it is possible that it is also the most relevant factor in other wireless sensor network operations such as routing.

5.2.2 Algorithm Comparison. AFL is found to use more energy by far. In every case it requires many more transmitted and received bits than Map Growing; it needs an average of almost 6 times as many transmitted bits and almost 4 times as many received bits. At the same time, AFL achieves a much better Average Distance Error than Map Growing. AFL's refinement estimates accurate positions, but requires much more communication to do so. Depending on the degree, a node can require between 700 and 2,500 iterations, broadcasting its position and receiving a broadcast from all neighbors during each iteration. On average Map Growing only transmits 35 messages while receiving 261. The extra communication helps AFL achieve half the Average Distance Error on average.

5.2.3 Energy Consumption Model. Using the experimental data, an Energy Consumption Model for both AFL and Map Growing is developed. The model is applied using data from the MICA2 and MICA2DOT wireless sensor nodes. The model estimates energy consumption given the node parameters, size and degree of the network, typical range errors, and deployment type. The model uses the AFL and Map Growing data, but since worst case estimates are used for processing and RF power consumption, the model can be applied to other incremental and concurrent algorithms as a rough, order of magnitude estimate.

5.2.4 Energy Finding. The most significant result of this research is wireless sensor network localization does not expend a significant amount of energy in a typical node. This indicates efforts to conserve communication costs in localization algorithms will not be as effective as hoped [IS03] [NN01] [SPS02] [TP03]. In the most extreme example (MICA2DOT, using a worst-case simulation and the worst case model), the energy expended by localization is 0.4%. 500mA-hr batteries are expected to last between 3-5 months under typical WSN conditions [Cro05]. 0.4% of 3 months (2,160 hours) accounts for about 8.64 hours of operation. So, in essence the potential for energy conservation in localization is at most 9 hours over 3 months.

5.3 Research Contributions

This research is the first known effort to evaluate the energy costs and error performance of wireless sensor network localization algorithms. It is the only (known) work that systematically quantifies and compares the energy costs for localization. Additionally, the resulting Energy Consumption Model for localization is an effective tool to predict the energy costs of other incremental and concurrent algorithms under a variety of conditions. The method used and presented also provides a framework to evaluate the energy costs of other areas of wireless sensor network operation. Using the presented method, energy consumption models may be developed for WSN routing, MAC protocols, target tracking and many other applications. In addition, the idea that degree most affects energy usage very likely extends to other node operations such as the WSN application, routing, and MAC protocols. Finally, the key energy finding that localization algorithms do not expend a significant amount of energy is an important result that can allow researchers to concentrate on other aspects of localization research such as accuracy, scalability, coverage and resilience.

5.4 Future Work

This research is limited in that only two algorithms are implemented, tested and modelled. Future research in this area should implement more algorithms and more types of algorithms to expand the Energy Consumption Model for Range Free algorithms and fully centralized algorithms. The behavior of AFL received bits exhibited two trends: 1) a drop when degree is increased from 8 to 12 because of quicker refinements with higher degrees, and 2) an increase when degree is increased from 12 to 16 because of the fact more nodes can hear each message. The behavior for the degree levels between 8 and 16 can be further investigated to find an optimal degree for AFL to operate at. This would ensure the best accuracy for the energy costs for that algorithm. Additionally, the energy costs due to processing is found to be significant, but is not examined closely. Future work may be able to more accurately investigate the processing energy costs involved and determine if it is beneficial to employ certain

speed optimizations to save energy at the cost of possible loss of accuracy. In terms of the algorithms themselves, improvements to both AFL and Map Growing can be investigated. For example, Map Growing does not use any neighbor data to restrict a position estimate to a bounded area. As a result, a node may accept a position that is far from its true position and far from its neighbors. Also, it is possible for the starting node in Map Growing to localize more of its neighbors instead of just two. This may reduce some error that gets propagated. For AFL, the number of messages needed in the reference node finding phase can be reduced. Only a subset of the nodes actually need to participate in the flooding to determine hopcounts to each reference node. After the phase is complete, nodes that did not participate only need to ask their neighbor that did and add 1 or -1 to each hopcount total to obtain their estimate. Since the initial estimate is already coarse, this approximation should not sacrifice accuracy but can save messages. Furthermore, this idea can extend AFL to be a simple but accurate localization algorithm for mobile wireless sensor nodes. Also, AFL exhibited the most error when nodes had few nodes. A modification can have low degree nodes try to use three neighbors to trilaterate or use the 2-Beacon Solver to find a position.

5.5 Summary

Wireless Sensor Networks are an exploding field of research, commerce and development. They can be found in an ever growing list of applications and fields. Since wireless sensor nodes are small and low powered with limited resources, significant work has been done to conserve the energy available in all phases of operation. Localization is critical to any data that the node provides for without location information, data can be useless [SRB01]. While accuracy has been an important measure of localization, energy consumption is also a localization concern. This work determines the single most important factor to localization energy consumption is node degree. The Energy Consumption Model can be used to estimate energy usage in a variety of situations for incremental and concurrent algorithms. Additionally, this research

presents a method used to quantify, evaluate and compare actual energy expended during localization and finds a significant amount of energy is not used during this process.

Appendix A. Graphs of Example Networks

This appendix contains graphs that represent the different types of networks that are used in this research. Not all graphs represent the x and y axis on the same scale. Lines between nodes indicate that the endpoints are within range of each other (15 meters) and are able to communicate.

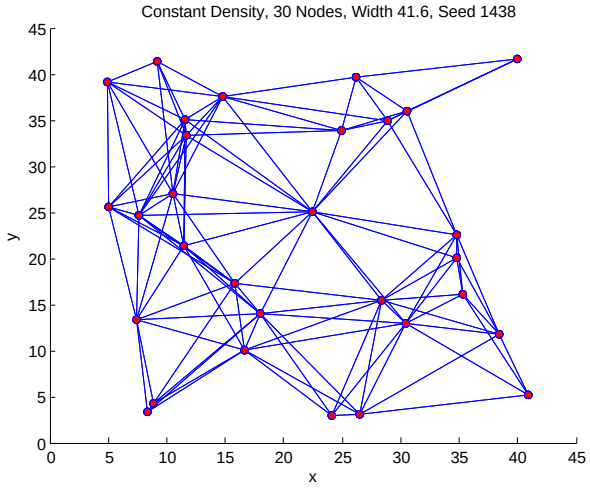


Figure A.1: 30 Nodes, Constant Density, Degree 8

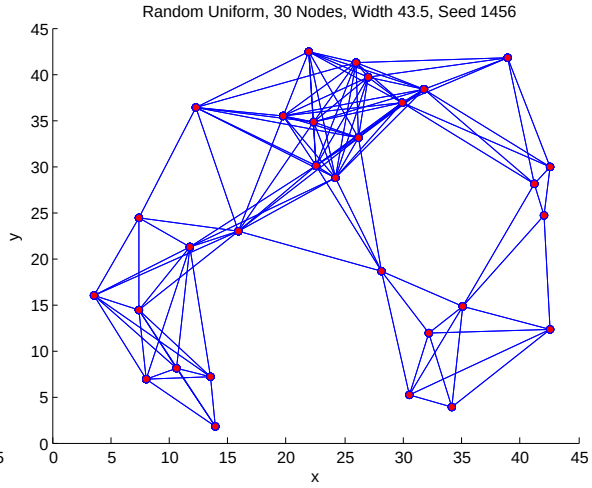


Figure A.2: 30 Nodes, Random Uniform, Degree 8

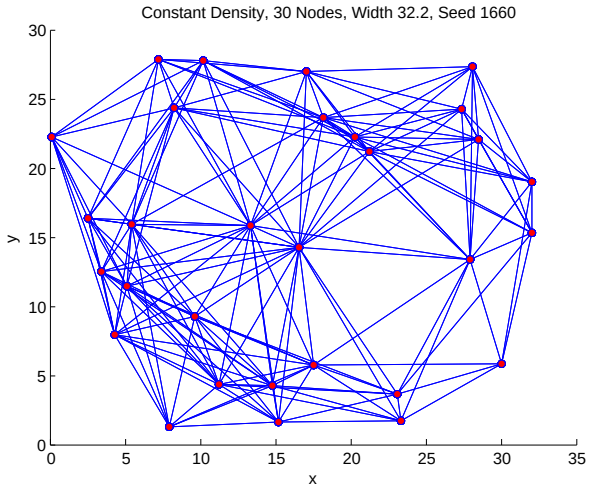


Figure A.3: 30 Nodes, Constant Density, Degree 12

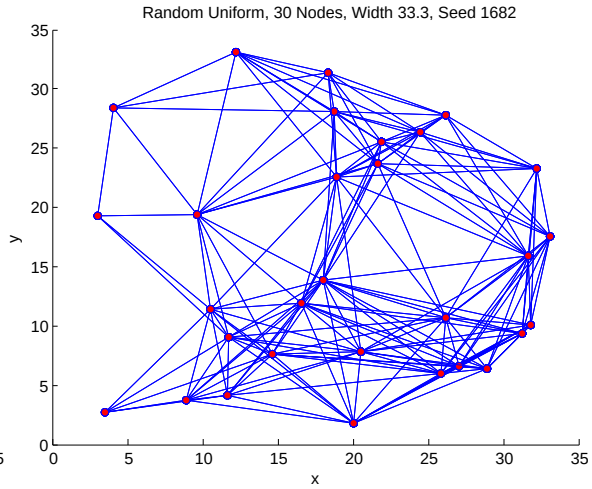


Figure A.4: 30 Nodes, Random Uniform, Degree 12

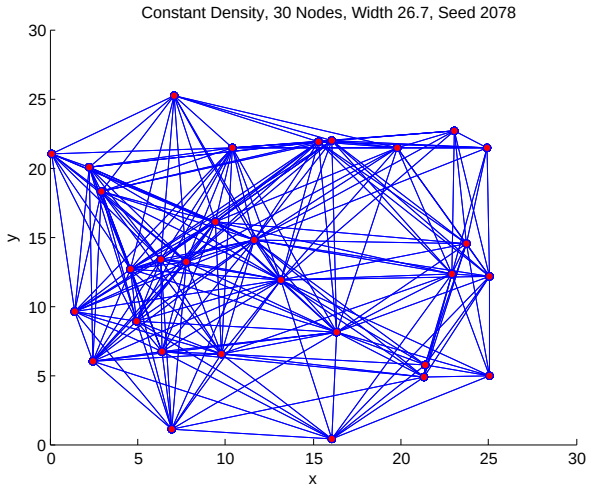


Figure A.5: 30 Nodes, Constant Density, Degree 16

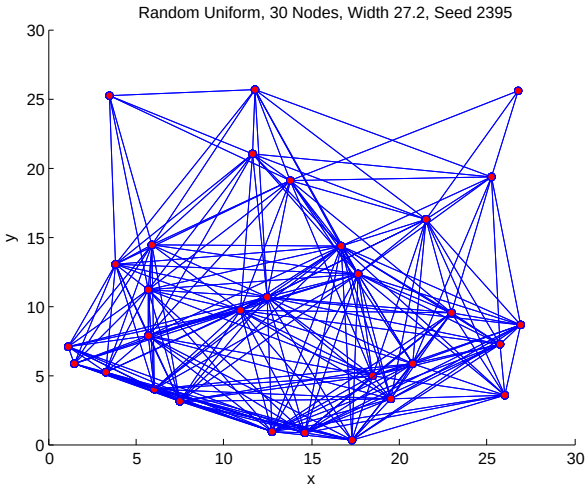


Figure A.6: 30 Nodes, Random Uniform, Degree 16

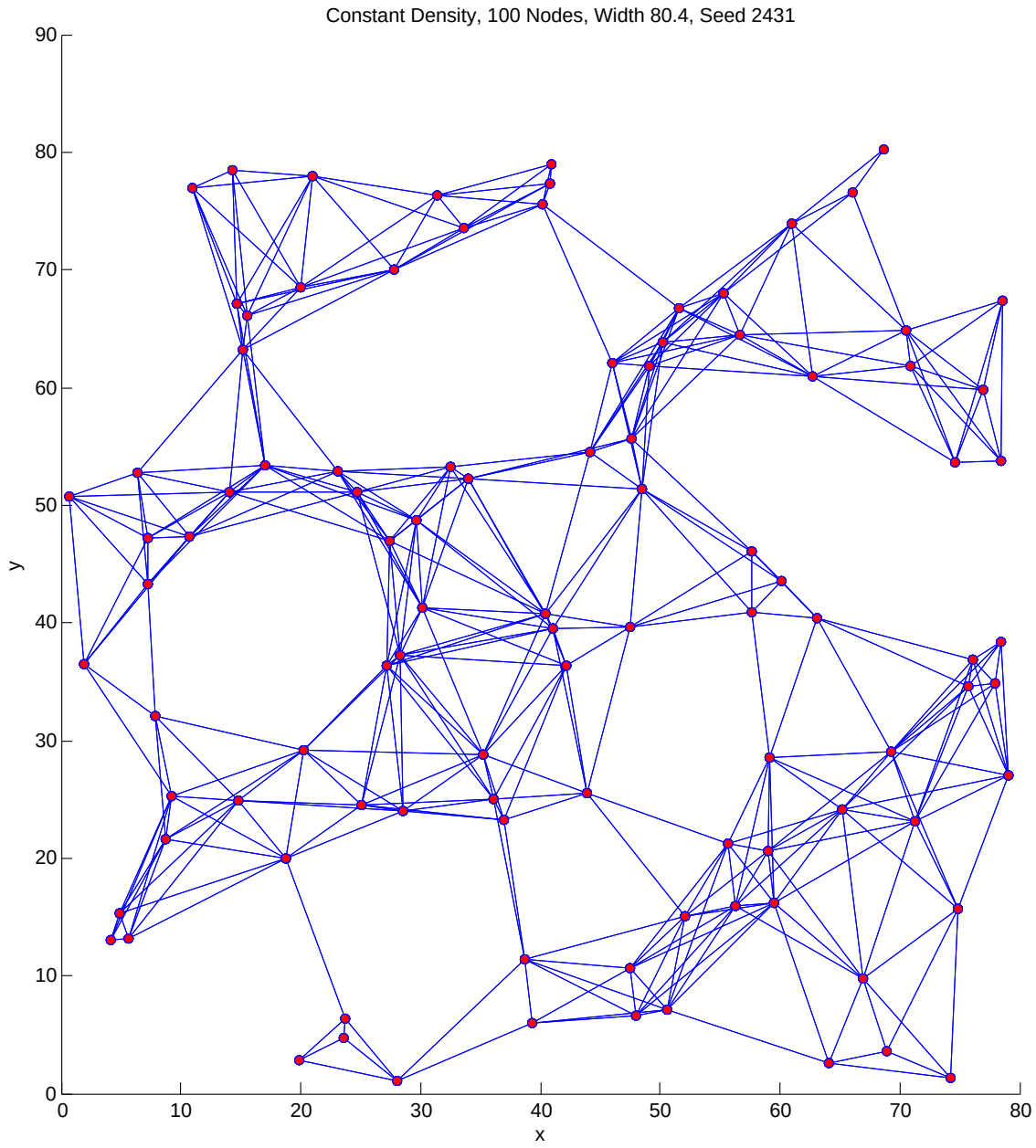


Figure A.7: 100 Nodes, Constant Density, Degree 8

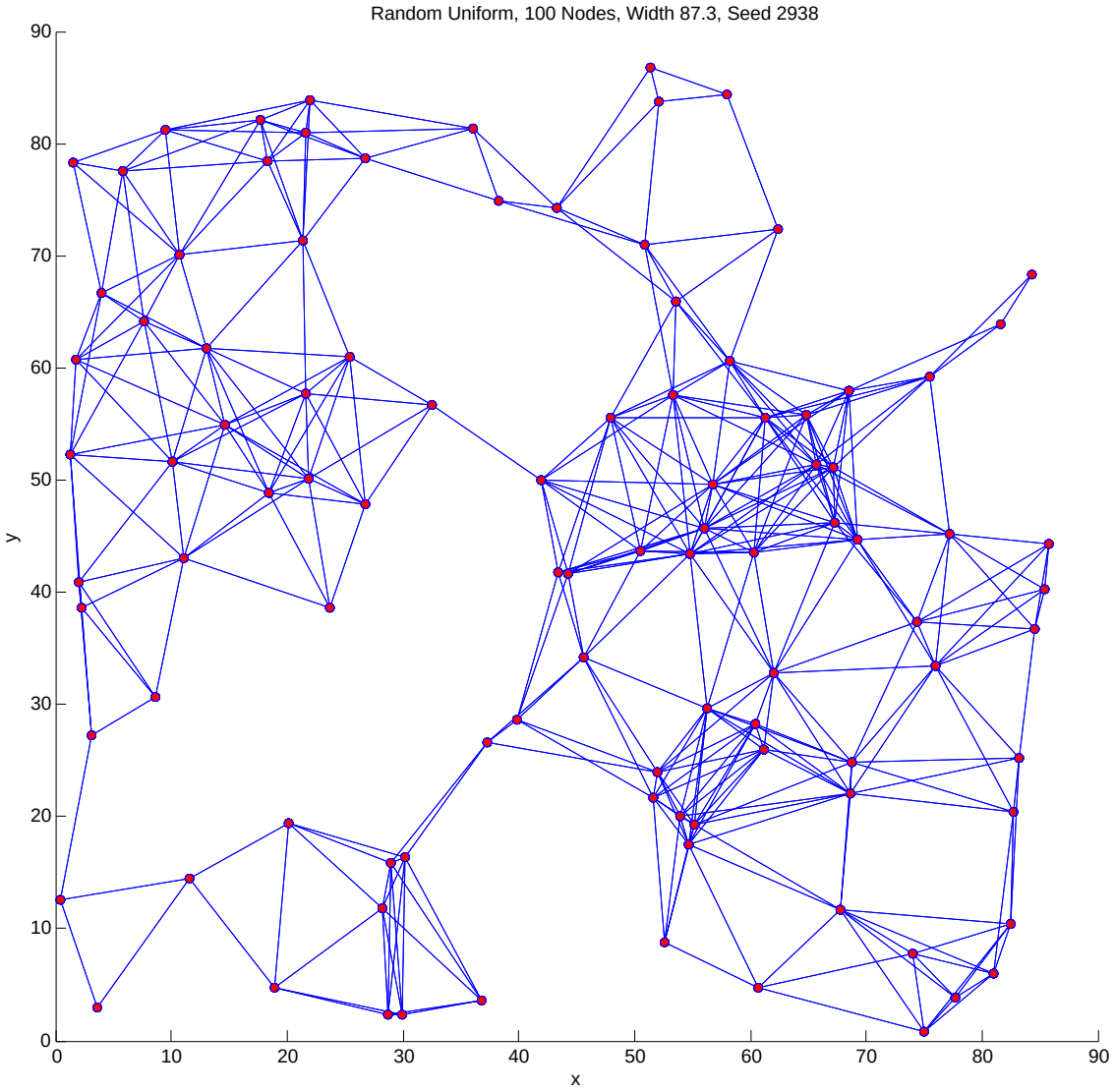


Figure A.8: 100 Nodes, Random Uniform, Degree 8

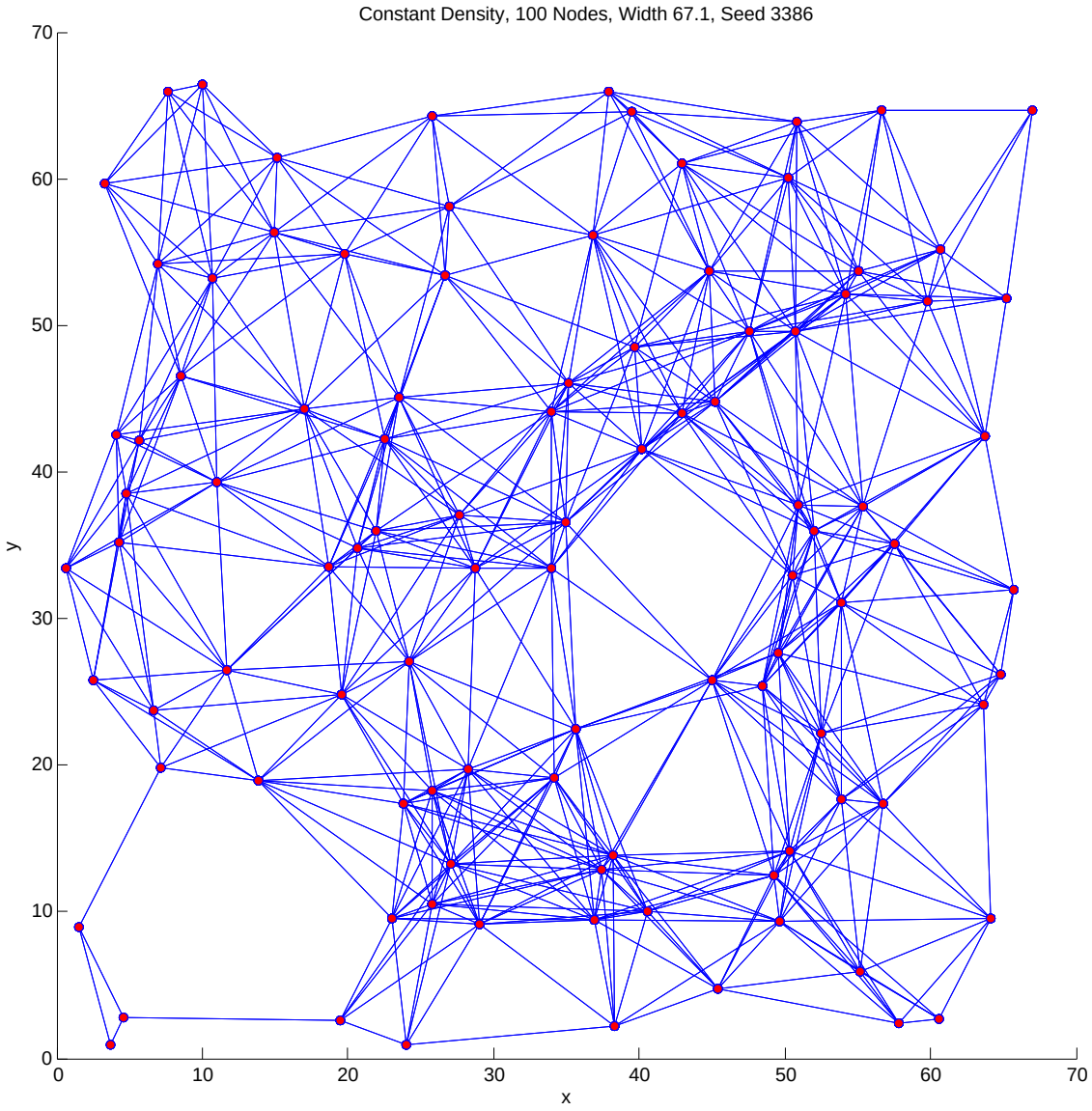


Figure A.9: 100 Nodes, Constant Density, Degree 12

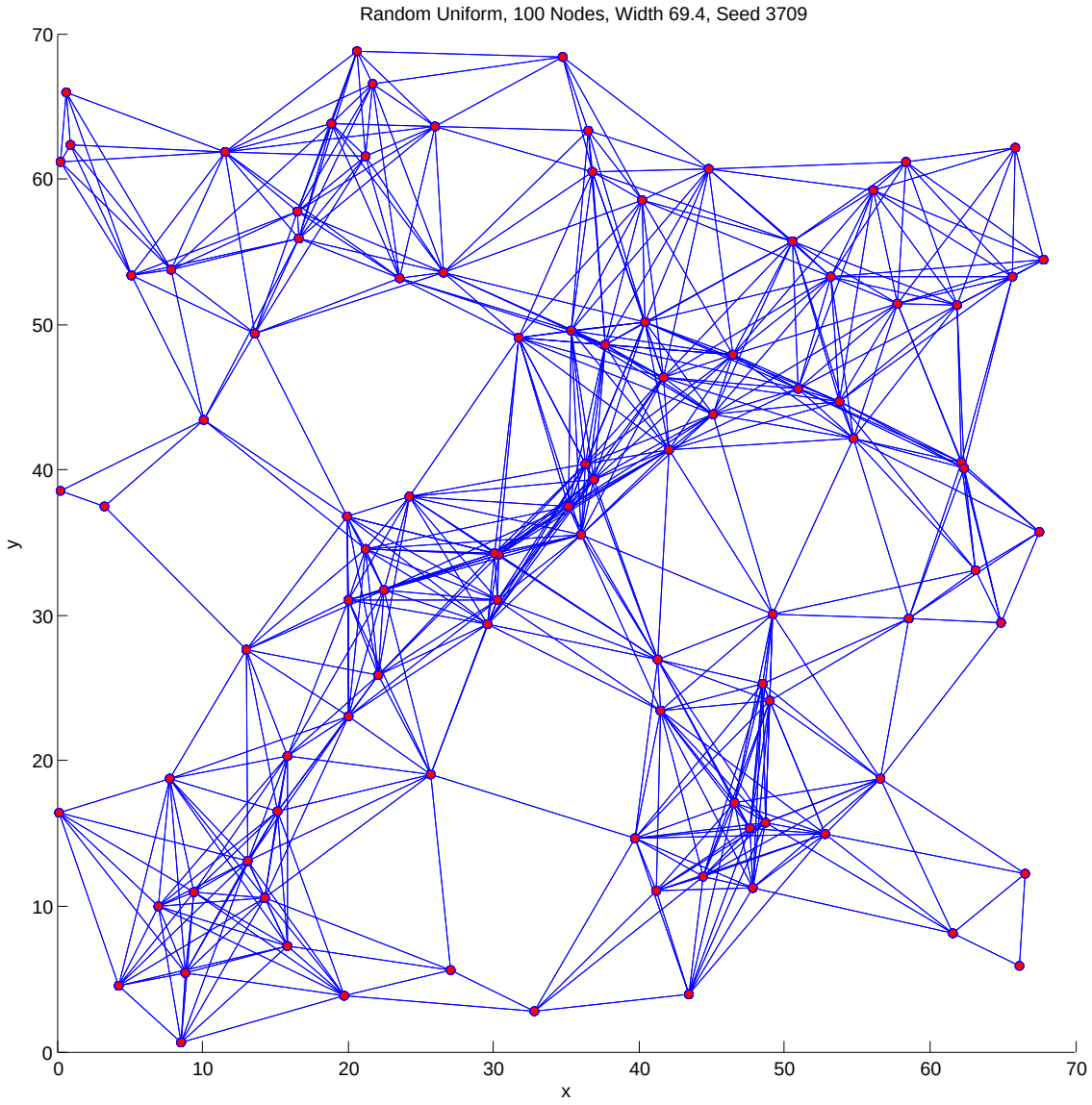


Figure A.10: 100 Nodes, Random Uniform, Degree 12

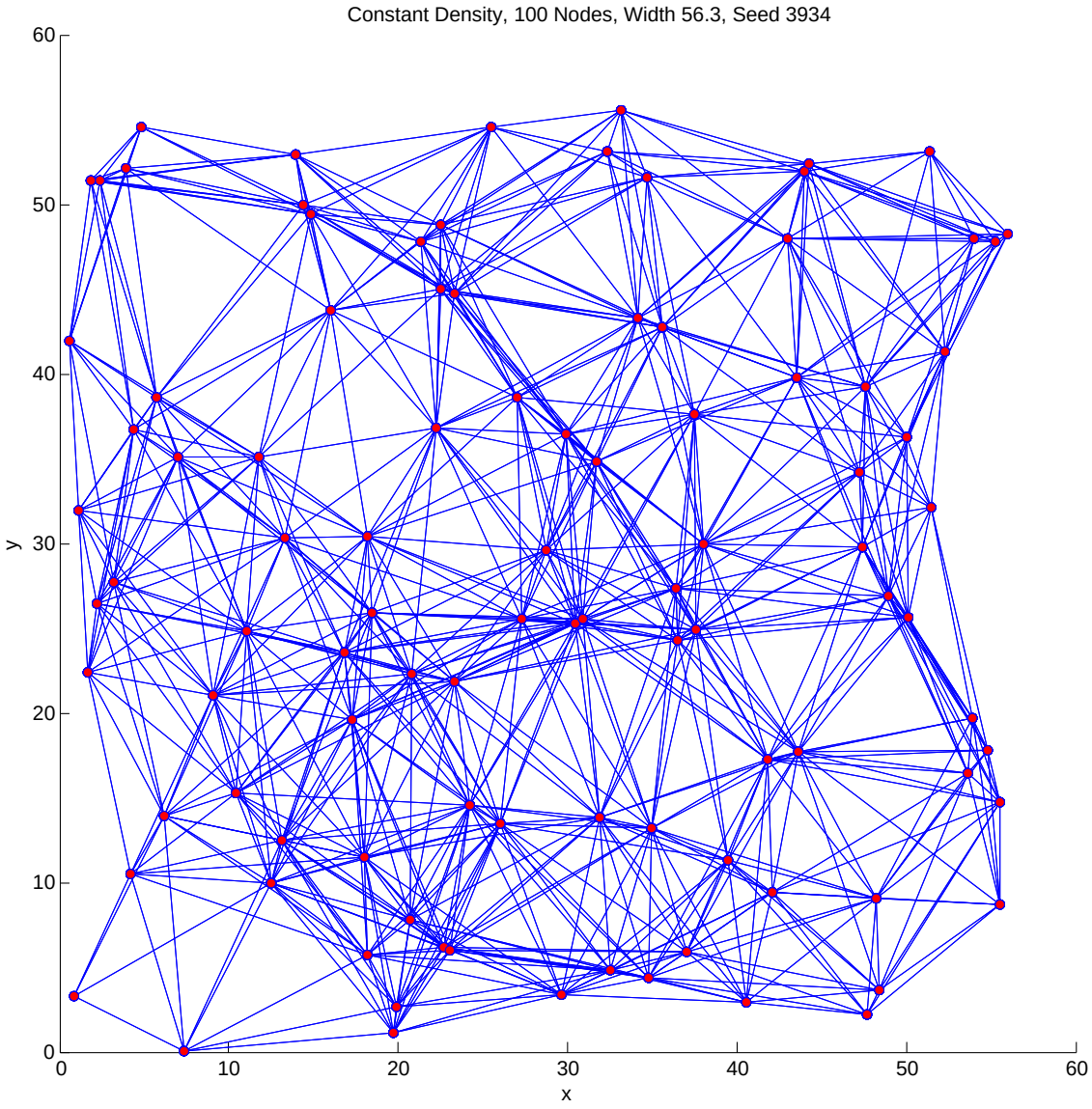


Figure A.11: 100 Nodes, Constant Density, Degree 16

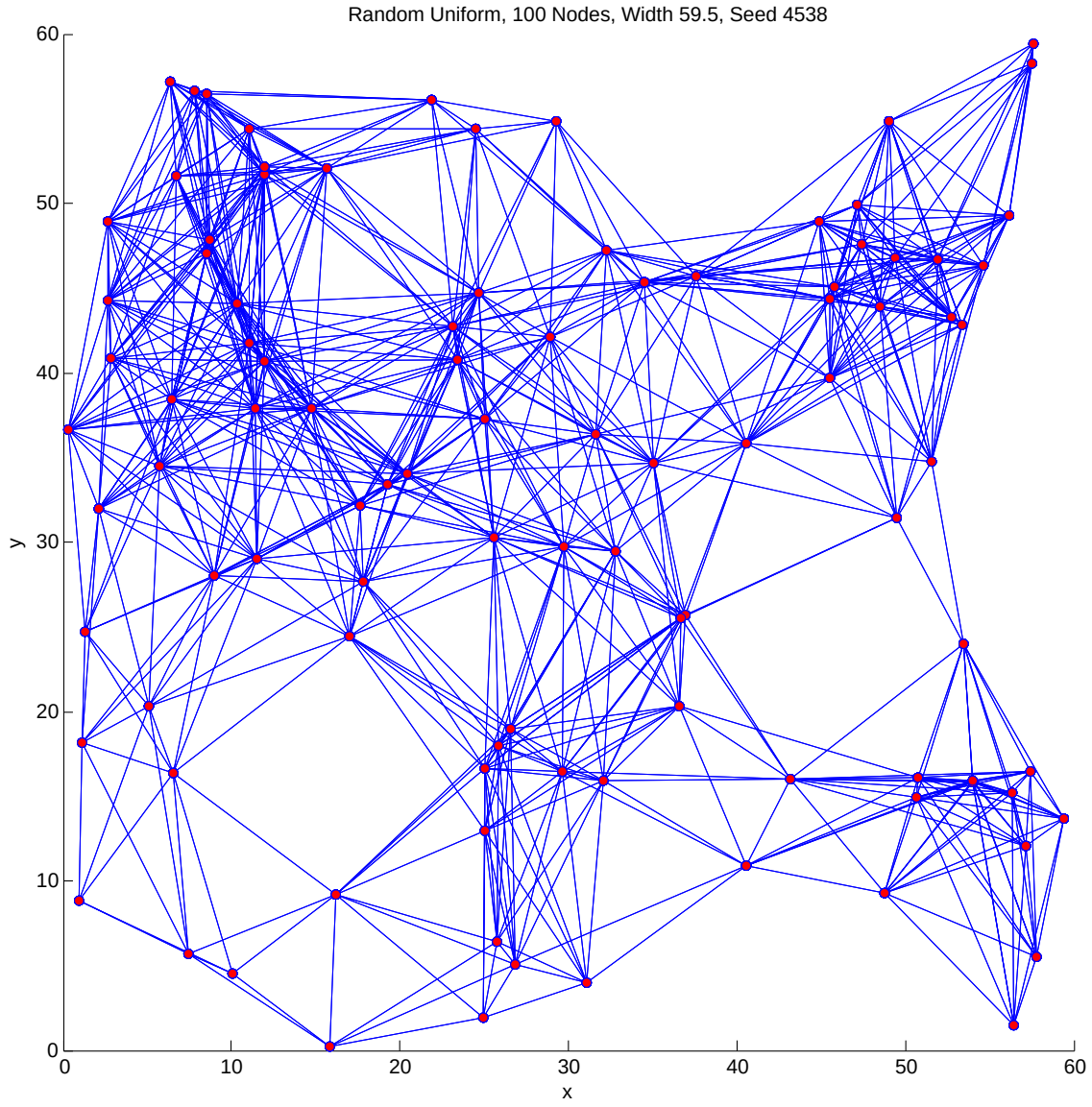


Figure A.12: 100 Nodes, Random Uniform, Degree 16

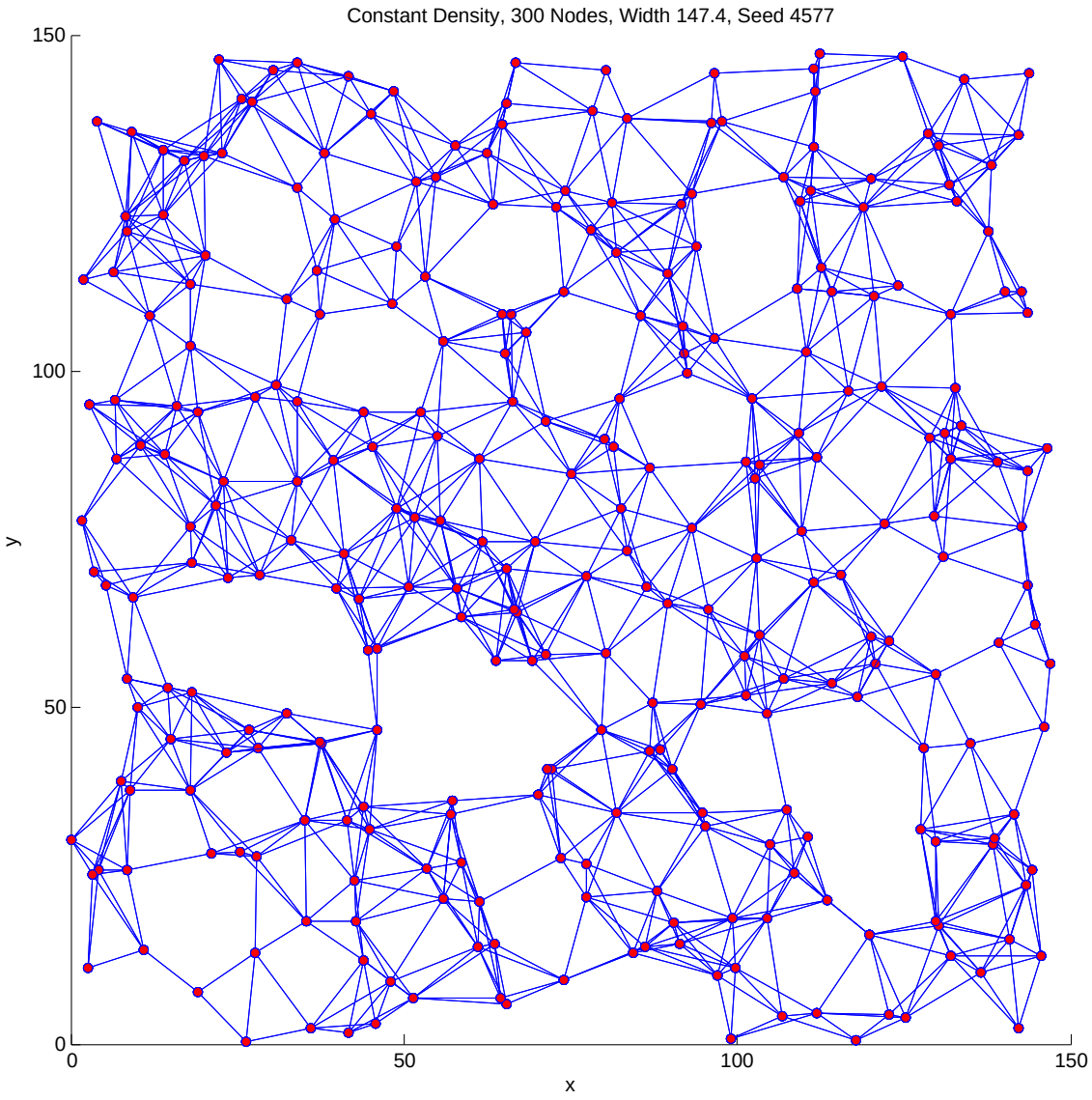


Figure A.13: 300 Nodes, Constant Density, Degree 8

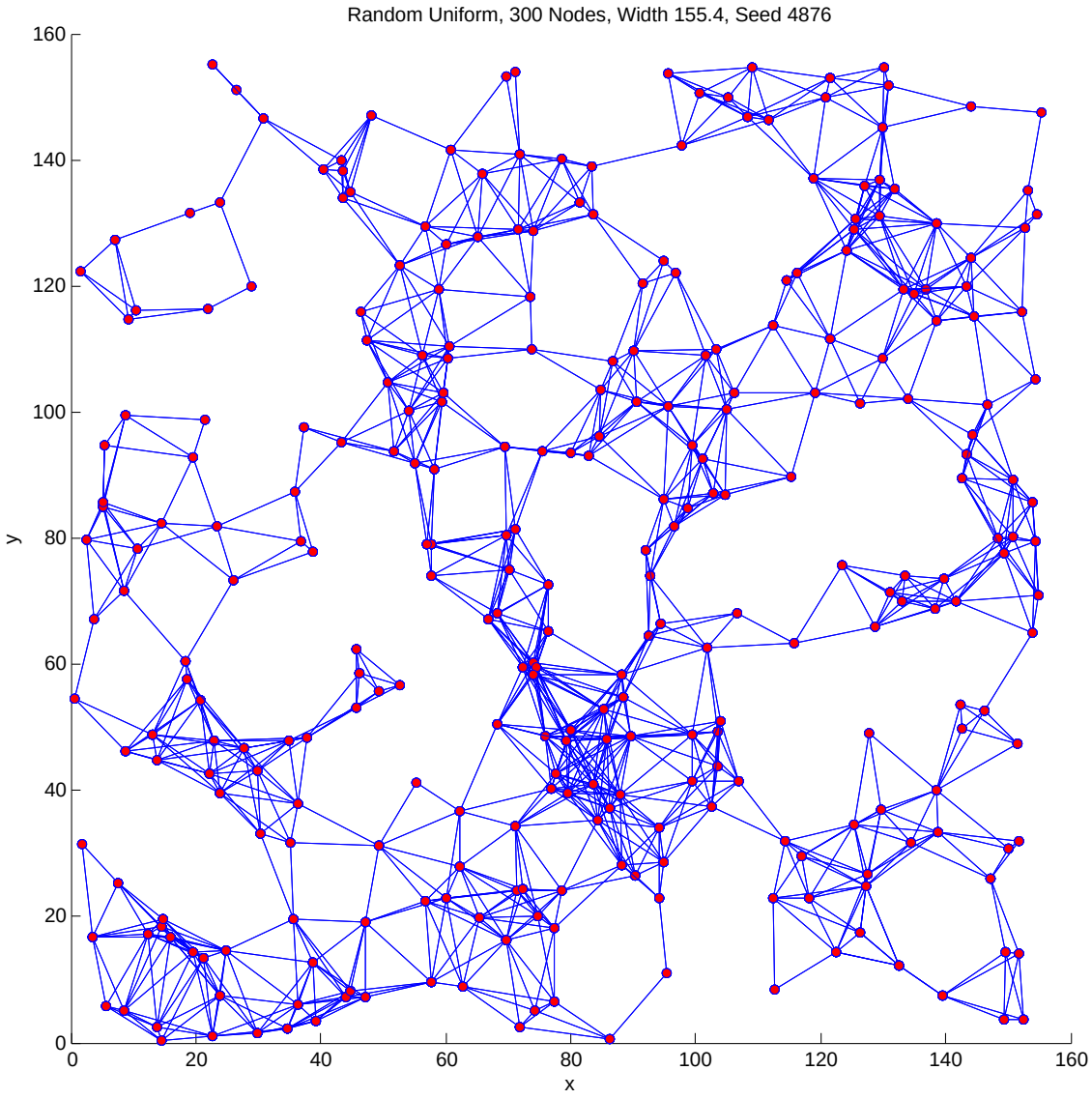


Figure A.14: 300 Nodes, Random Uniform, Degree 8

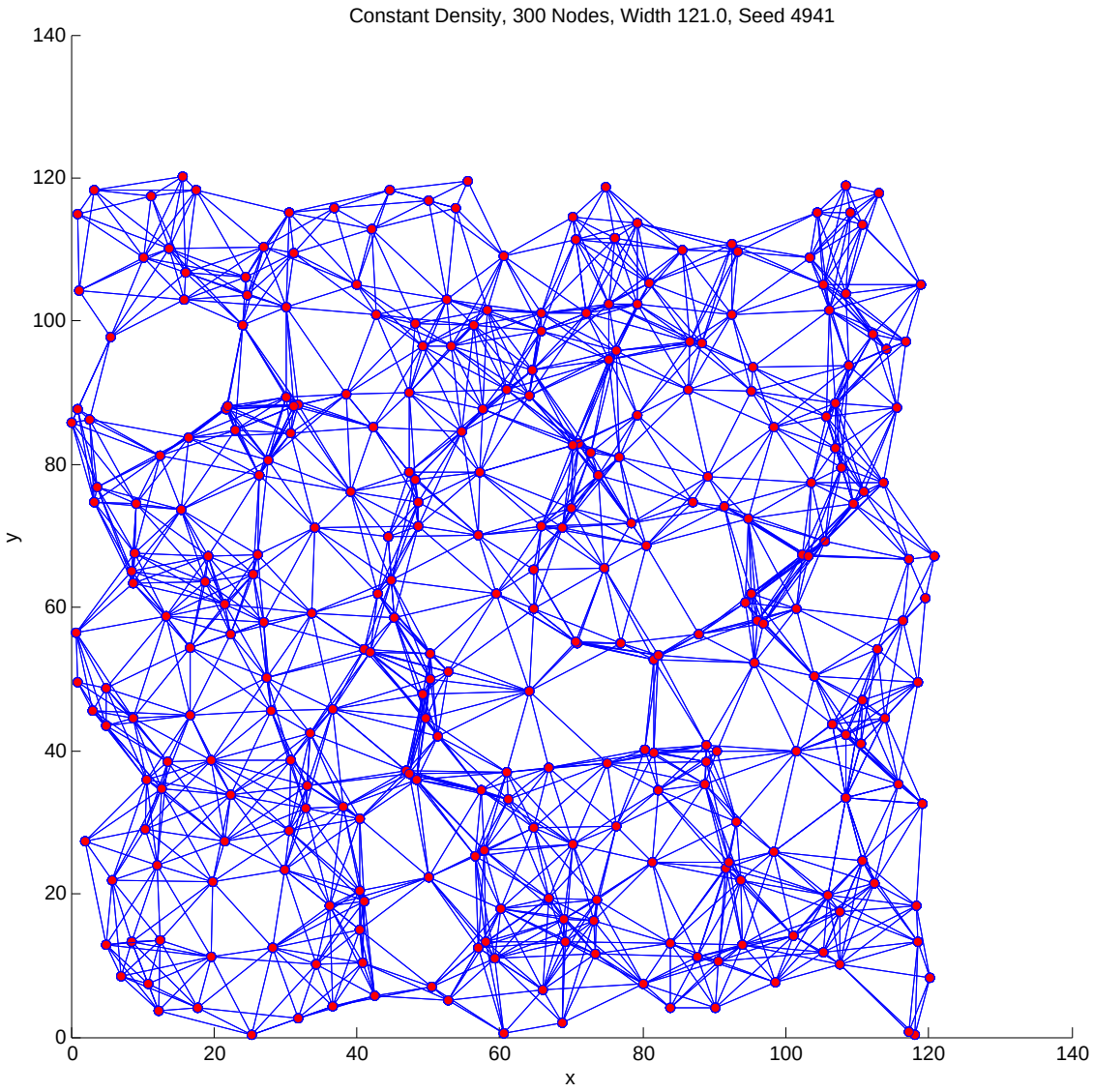


Figure A.15: 300 Nodes, Constant Density, Degree 12

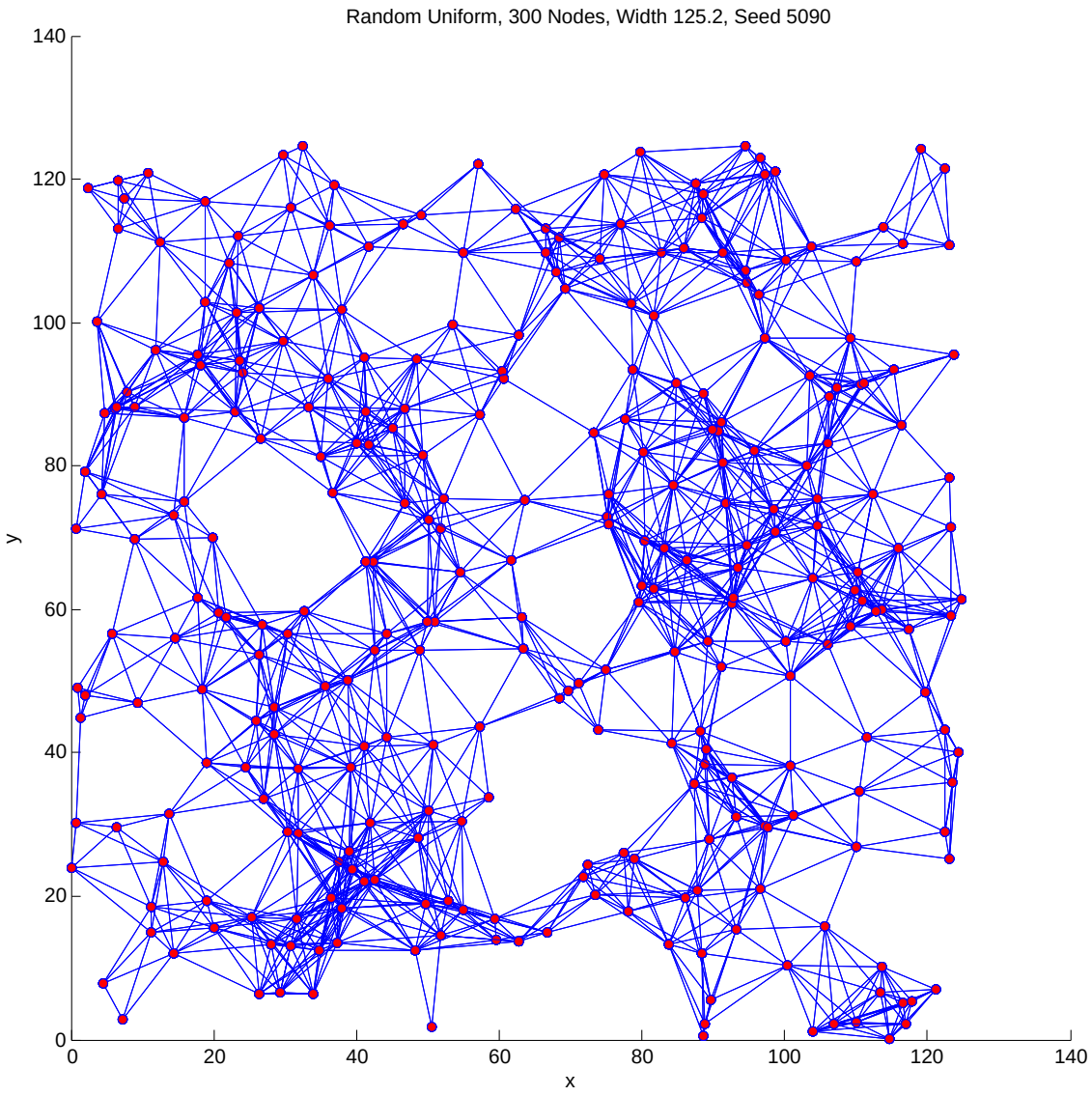


Figure A.16: 300 Nodes, Random Uniform, Degree 12

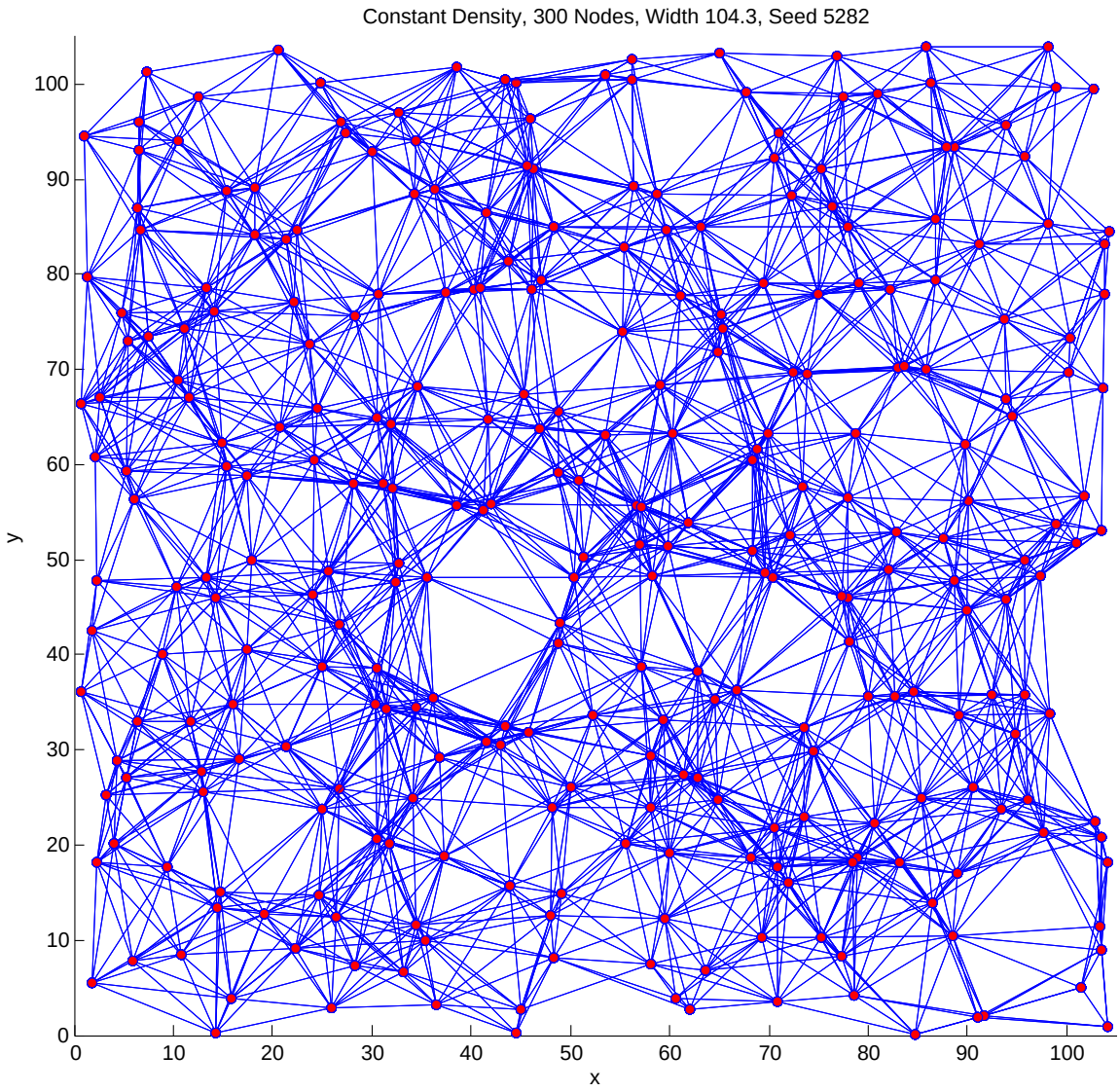


Figure A.17: 300 Nodes, Constant Density, Degree 16

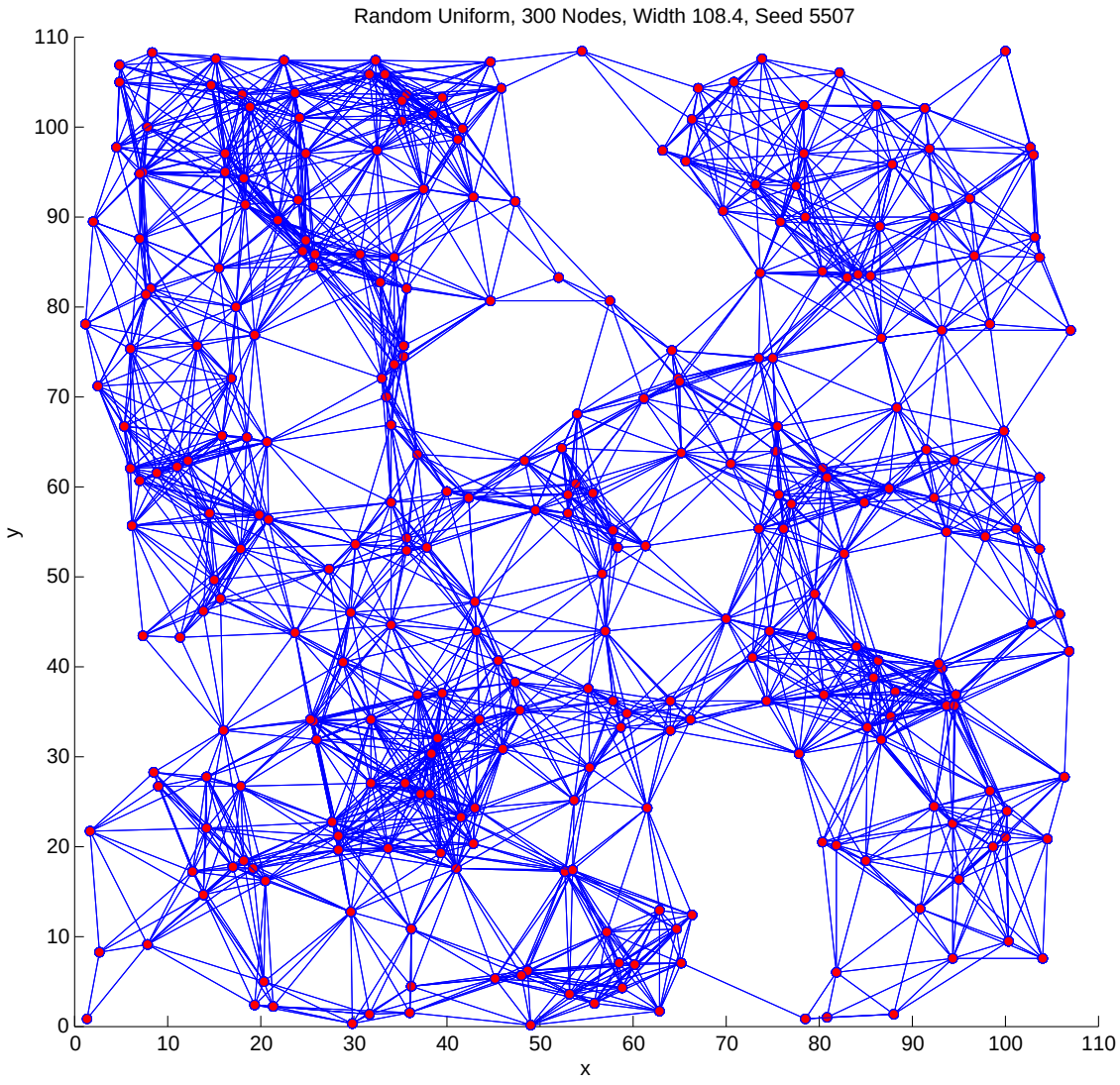


Figure A.18: 300 Nodes, Random Uniform, Degree 16

Appendix B. Lateration Method

The following describes the lateration method for an unknown node to estimate its position using the known positions of 3 neighbors and distance estimates to each neighbor. This method is adapted from [LR03].

From the estimated distances, d_i, to each localized neighbor and the positions of each neighbor (x_i, y_i), the following system of equations is derived. The unknown position is denoted as (x, y).

$$(x_1 - x)^2 + (y_1 - y)^2 = d_1^2 \tag{B.1}$$

$$(x_2 - x)^2 + (y_2 - y)^2 = d_2^2 \tag{B.2}$$

$$(x_3 - x)^2 + (y_3 - y)^2 = d_3^2 \tag{B.3}$$

The system is linearized by subtracting the last equation from the first 2 equations.

$$x_1^2 - x_3^2 - 2(x_1 - x_3)x + y_1^2 - y_3^2 - 2(y_1 - y_n)y = d_1^2 - d_3^2 \tag{B.4}$$

$$x_2^2 - x_3^2 - 2(x_2 - x_3)x + y_2^2 - y_3^2 - 2(y_2 - y_n)y = d_2^2 - d_3^2 \tag{B.5}$$

The terms are reordered to create a proper system of linear equations in the form $Ax = b$.

$$A = \begin{bmatrix} 2(x_1 - x_3) & 2(y_1 - y_3) \\ 2(x_2 - x_3) & 2(y_2 - y_3) \end{bmatrix}$$

$$b = \begin{bmatrix} x_1^2 - x_3^2 + y_1^2 - y_3^2 + d_3^2 - d_1^2 \\ x_2^2 - x_3^2 + y_2^2 - y_3^2 + d_3^2 - d_2^2 \end{bmatrix}$$

The system is solved using a standard least-squares approach: $\hat{x} = (A^T A)^{-1} A^T b$. In rare cases, the matrix is not invertible and the lateration fails. An additional check can be used to verify that the result is consistent. The residue between the given distances, d_i, and the distance to the location estimate $\hat{x}$ is calculated for this check.

$$\text{residue} = \frac{\sum_{i=1}^{3} \left(\sqrt{(x_i - \hat{x})^2 + (y_i - \hat{y})^2} - d_i \right)}{3} \tag{B.6}$$

In essence, this is the average difference between the given distances and the distances resulting from the lateration. Large residue indicates an inconsistent set of equations. The location $\hat{x}$ should be rejected if the residue exceeds the radio range.

Appendix C. Validation Figures for OPNET-AFL

This appendix includes the figures comparing the OPNET-AFL against the published results for degrees 5 to 13 at fractional range errors 0.01 to 0.07. OPNET-AFL is plotted with 95% confidence intervals on the 10 repetitions of each experiment.

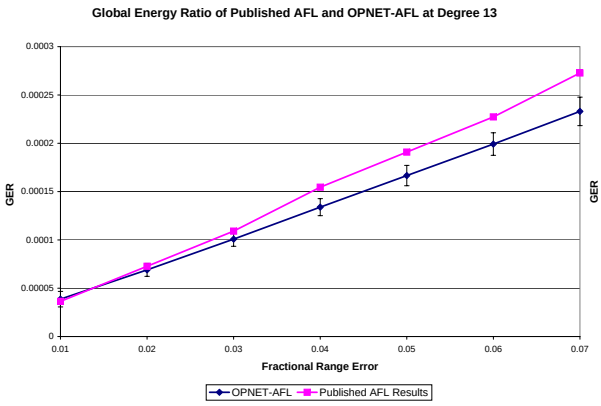


Figure C.1: OPNET-AFL and published AFL results, avg degree = 13

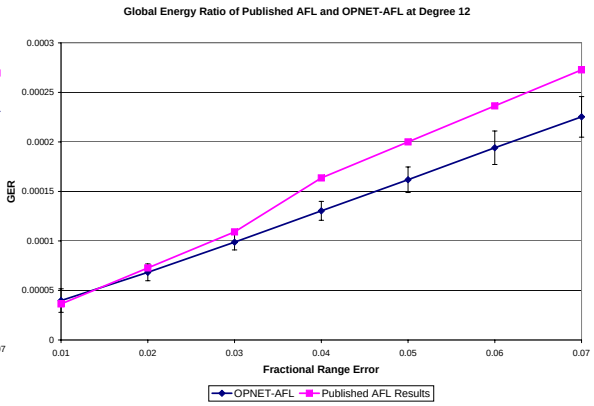


Figure C.2: OPNET-AFL and published AFL results, avg degree = 12

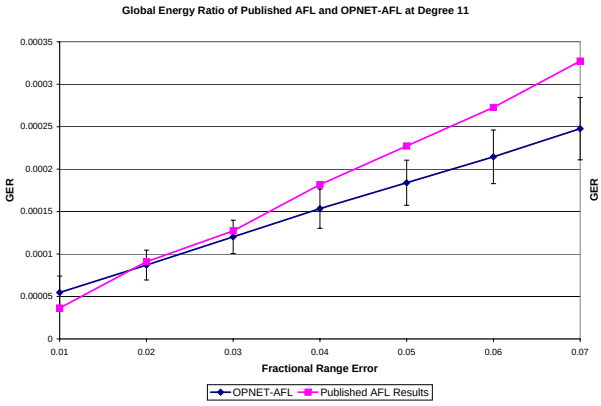


Figure C.3: OPNET-AFL and published AFL results, avg degree = 11

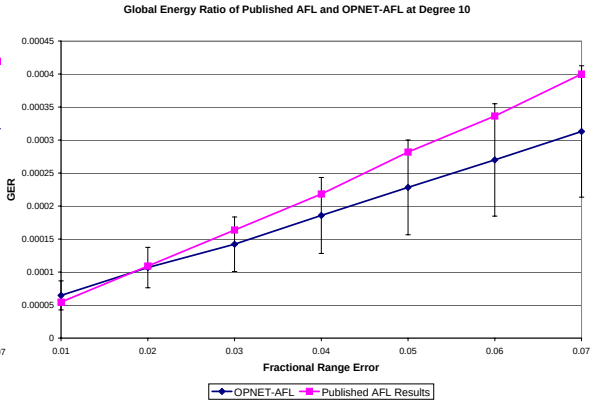


Figure C.4: OPNET-AFL and published AFL results, avg degree = 10

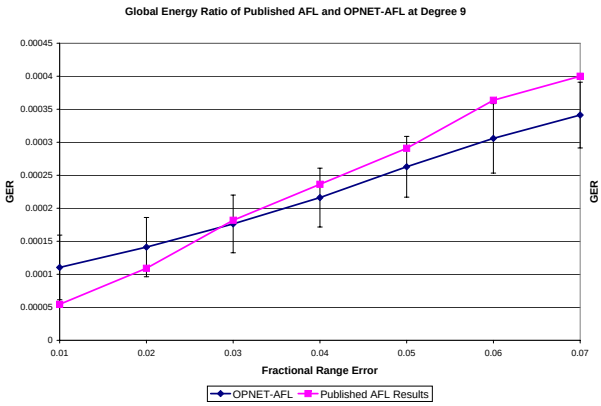


Figure C.5: OPNET-AFL and published AFL results, avg degree = 9

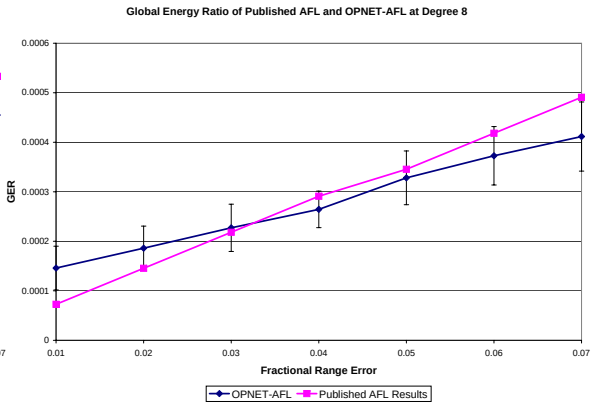


Figure C.6: OPNET-AFL and published AFL results, avg degree = 8

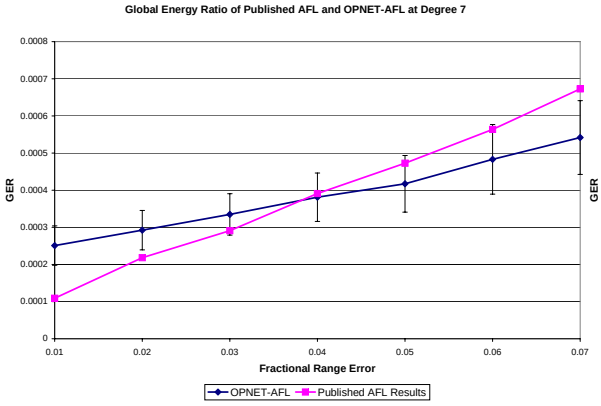


Figure C.7: OPNET-AFL and published AFL results, avg degree = 7

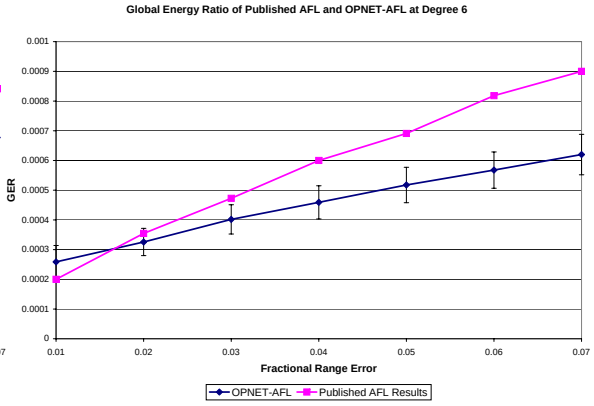


Figure C.8: OPNET-AFL and published AFL results, avg degree = 6

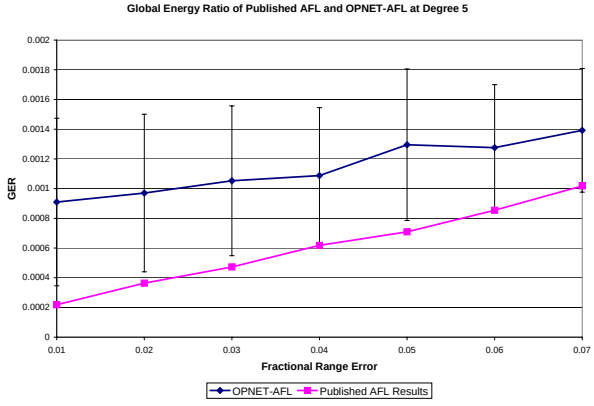


Figure C.9: OPNET-AFL and published AFL results, avg degree = 5

Appendix D. Network Generation Perl Script

This appendix lists the Perl code that created all the Constant Density and Random Uniform networks used in this research.

getmygraphs3.pl

```
#!/usr/local/ActivePerl-5.6/bin/perl -w
#getmygraphs3.pl
#takes a file of carriage return delimited random numbers
#creates 18 networks for each random number
#All combinations of size 30,100, 300, degree 8,12,16,
#and type constant density (CD) and random uniform (RU).
if(@ARGV>1){
    $seeds = $ARGV[0];
    $thresh = $ARGV[1];
}else {
    print "give me a file of random numbers\n";
    exit(0);
} open(NUMS, $seeds) or die "can't read ".$seeds; @lines = <NUMS>;
close(NUMS); foreach $line (@lines){
    chop($line);
    chop($line);
    for($i = 2; $i < 5; $i++)
    {
        @lista = ("perl", "graphgen3.pl", 30 , $i*4, "CD", $line, $thresh);
        system (@lista)  == 0 or die "system @lista failed: $?";
        @listb = ("perl", "graphgen3.pl", 30 , $i*4, "RU", $line, $thresh);
        system (@listb)  == 0 or die "system @listb failed: $?";
        @listc = ("perl", "graphgen3.pl", 100 , $i*4, "CD", $line, $thresh);
        system (@listc)  == 0 or die "system @listc failed: $?";
        @listd = ("perl", "graphgen3.pl", 100 , $i*4, "RU", $line, $thresh);
        system (@listd)  == 0 or die "system @listd failed: $?";
        @liste = ("perl", "graphgen3.pl", 300 , $i*4, "CD", $line, $thresh);
        system (@liste)  == 0 or die "system @liste failed: $?";
        @listf = ("perl", "graphgen3.pl", 300 , $i*4, "RU", $line, $thresh);
        system (@listf)  == 0 or die "system @listf failed: $?";
    }
}
#------------end getmygraphs3.pl-------------------------
```

graphgen3.pl

```
#!/usr/local/ActivePerl-5.6/bin/perl -w
#
#graphgen3.pl - takes number of nodes, target degree, nw type, seed and threshold
# and produces a graph within threshold of the target degree.
# Also outputs a file that can be used in Matlab to create a graph of the network.
# Also generates a connected.csv that identifies the graph file,
# and tells whether its connected or not and the actual average degree
# v2 - Makes sure CD/RU graphs are connected
# v3 - better uses makemfile and allows range about target degree to be editable
if(@ARGV>4){
    $numnodes = $ARGV[0];
    $target = $ARGV[1];
    $type = $ARGV[2];
    $seed = $ARGV[3];
    $thresh = $ARGV[4];
    if(!($type eq "RU") && !($type eq "CD"))
        {
    print "Valid types are RU and CD\n";
    exit(0);
    }
}else {
    print "give me a number of nodes, target degree, nw type, and seed\n";
    exit(0);
} #$mfile = $numnodes."."$target."."$type."."$seed."."m";

%pos = ();          #create a hash used to store positions and node ids.
srand($seed);       #seed the PRNG
$avgdeg = $target - 3;          #while the average degree is far from the target
while ( $avgdeg < ($target-$thresh) || $avgdeg > ($target+$thresh)) {
    if ($type eq "RU"){
    if($numnodes == 300){ #equations based on regressions to calculate network width
        $width = -0.0514*$target*$target*$target
                 + 2.2657*$target*$target - 37.218*$target + 334.41; }
    if($numnodes == 100){
        $width = -0.0307*$target*$target*$target
                 + 1.3565*$target*$target - 22.286*$target + 194.51;}
    if($numnodes == 30){
        $width = -0.0134*$target*$target*$target
                + 0.6121*$target*$target - 10.716*$target + 96.872;}
    if($numnodes != 300 && $numnodes != 100 && $numnodes != 30){
        print "I can only handle 300, 100 or 30 right now\n";
        exit(0);}
    #print "$width\n";
    $nn = sprintf("%03i", $numnodes);
    $w =  sprintf("%02.1f", $width);
    $s = sprintf("%09i", $seed);
    $pfile = "2".$nn.$w.$s."."pos";   #build the file name of the position file
    $connected = 0;
    while($connected == 0){
        for($i = 0; $i<$numnodes; $i++)
            {
                $namex = $i."x";        #for each node, randomly
                $namey = $i."y";        #select an x and y anywhere in network
                $currx = rand($width);
                $curry = rand($width);
                $pos{$namex}= $currx;
                $pos{$namey}= $curry;
            }
        open(POUT, ">$pfile"); #print $pfile, "\n";
        for($i = 0; $i < $numnodes; $i++)
            {                           #write positions to a file
            print POUT "$i,".$pos{$i."x"}.",".$pos{$i."y"}."\n";
            }
        close(POUT);              #call consoleconntest.pl to see if its connected
```

```
    $response = `perl consoleconntest.pl $pfile`;
    print $response;        #and learn the average degree of the network
    @resplist = split(',', $response);
    $connected = $resplist[1];
    $avgdeg = $resplist[2];
    }
}
if ($type eq "CD")
{
 if($numnodes == 300){  #equations based on regressions to calculate network width
     $width = -0.0659*$target**3 + 2.6785*$target**2
          - 40.138*$target + 330.78;}
 if($numnodes == 100){
     $width = -0.068*$target**3 + 2.5276*$target**2
          - 33.219*$target + 219.23;}
 if($numnodes == 30){
     $width = -0.0247*$target**3 + 1.0128*$target**2
          - 15.104*$target + 110.25;   }
 $nn = sprintf("%03i", $numnodes);
 $w =  sprintf("%02.1f", $width);
 $s = sprintf("%09i", $seed);
 $pfile = "5".$nn.$w.$s."."pos";   #build the position file name
 $side = int(sqrt($numnodes/3));    #number of squares on one side
 $unit = $width/$side;              #width of one square
 $unitset = $numnodes/($side*$side);#Number of nodes in one square
 $connected = 0;
 while ($connected == 0)             #ensure network is connected
    {
        for($i = 0; $i<$numnodes; $i++)
            {
            if($i < $unitset*$side*$side && $i != 0) #put 0 in a random place
            {                                #all others are placed in their
                $minx = (int($i/($unitset))%$side)*$unit;
                $maxx = $minx+$unit;#corresponding square based on their node id
                $miny = int($i/($unitset*$side))*$unit;
                $maxy = $miny+$unit;
                if((($i+1)%$unitset)== 0)
                {
                    if($unit > 15)  #if the side of a square is bigger than the
                    {           #range of the node, one node in square is placed
                        $minx = $minx+($unit-15+.5);#in an area
                        $miny = $miny+($unit-15+.5);#to promote connectivity
                    }
                }
                if($i != 0 && $i != 1)     #for nodes with id > 1, make sure they
                {                       #are connected to at least one other node
                    $tempdeg = 0;
                    while($tempdeg == 0 )
                    {
                        $namex = $i."x";
                        $namey = $i."y";
                        $currx = $minx + rand($maxx - $minx);
                        $curry = $miny + rand($maxy - $miny);
                        $pos{$namex}= $currx;
                        $pos{$namey}= $curry;
                        $tempdeg = 0;
                        for($j=0; $j<$i || $tempdeg == 0; $j++)
                        {
                            $dist = sqrt(($pos{$j."x"}-$currx)**2
                                 +($pos{$j."y"}-$curry)**2 );
                            if($dist<=15.0)
                            {
                                $tempdeg++;
                            }
                        }
                    }
```

```
                    }
                    else
                    {
                        $namex = $i."x";
                        $namey = $i."y";
                        $currx = $minx + rand($maxx - $minx);
                        $curry = $miny + rand($maxy - $miny);
                        $pos{$namex}= $currx;
                        $pos{$namey}= $curry;
                    }
                }
                else
                {
                    $namex = $i."x";
                    $namey = $i."y";
                    $currx = rand($width);
                    $curry = rand($width);
                    $pos{$namex}= $currx;
                    $pos{$namey}= $curry;
                }
            }
            open(POUT, ">$pfile"); #print $pfile, "\n";
            for($i = 0; $i < $numnodes; $i++)
            {                               #write positions to a file
                print POUT "$i,".$pos{$i."x"}.",".$pos{$i."y"}."\n";
            }
            close(POUT);        #call consoleconntest.pl to see if its connected
            $response = `perl consoleconntest.pl $pfile`;
            print $response;    #and learn the average degree of the network
            @resplist = split(',', $response);
            $connected = $resplist[1];
            $avgdeg = $resplist[2];
        }
     }
}
print "SUCCESS!! ";         #call conntest.pl to write connectivity
@conncall = ("perl", "conntest.pl", $pfile); #and degree to a separate file
system (@conncall)  == 0 or die "system @conncall failed: $?";
$degresponse = `perl consolemakemfile.pl $pfile`;  #call consolemakemfile.pl to
print $degresponse;                                #write Matlab graph to a file
#------------end graphgen3.pl-------------------------
```

conntest.pl (consoleconntest.pl is the same script, except the connectedness and average degree are written to the command line)

```
#!/usr/local/ActivePerl-5.6/bin/perl -w
# conntest.pl
# takes a position file and outputs a file
# that says if it is connected or not
# and the average degree

if(@ARGV>0){
  $input = $ARGV[0];
}else {
  print "give me a file name\n";
  exit(0);
}
#$output = $input;
if($input =~ /\/|\\/){ #I have a file in another directory
    @dirs = split /\/|\\/, $input;
    $numdirs = @dirs;
    $dirs[$numdirs-1] = "connected.csv";
    $output = "";
    for($t = 0; $t< $numdirs-1; $t++){
```

```
    $output = $output.$dirs[$t]."\\";
    }
    $output = $output.$dirs[$numdirs-1];
}else {
    $output ="connected.csv";
}
%actual = ();                                   #create hash
@names = ();                                       #create array
open(INFILE, $input) or die "can't read ".$input;
@lines = <INFILE>;                                 #read in all lines of file
close(INFILE);
$i = 0;
$length = @lines;
@tmp = ();              #temporary array to store a list of a nodes neighbors
@conn = ();             #a 2 dimensional array to store all the list of neighbors
@reach = ();            #an array of nodes that 0 can reach
$reached = 0;           #the number that 0 can reach
@test = split(',', $lines[0]);
if(!($test[0] eq "0")){
    print "$input not formatted correctly: word1: $test[0]\n";
    exit 0;
}
print "$input -> $output\n";
if ( open (OUTFILE, $output) ){
    close (OUTFILE);
    open(OUTFILE, ">>$output");
} else{
    close (OUTFILE);
    open(OUTFILE, ">>$output");
    print OUTFILE "Filename, Connected, AvgDeg\n";
}

open(OUTFILE, ">>$output");
foreach $line (@lines){                 #loop through all lines
    $linenum++;
    @words = split(',', $line);
    #grab the data
    $name = $words[0];           #first is node id
    $actx = $words[1];
    $acty = $words[2];
    $nameactx = $name."actx";
    $nameacty = $name."acty";
    $actual{$nameactx}=$actx;
    $actual{$nameacty}=$acty;
    $names[$i]=$name;
    $i++;
    if($linenum == $length){
    #determine if network is connected. ###############
    #first store connecitivity
    $deg = 0;
    for($j = 0; $j<$i; $j++){
        $n = 0;
        for($k=0; $k<$i; $k++){
        $dij = sqrt(($actual{$j."actx"}-$actual{$k."actx"})**2+
                    ($actual{$j."acty"}-$actual{$k."acty"})**2 );
        if($dij <= 15){
            $tmp[$n]=$k;
            $n++;
            if($j!=$k){$deg++;}
            }
        }
        $conn[$j]=[ @tmp ];
        @tmp = ();
    }
    $avgdeg = $deg/$i;
    #Can node 0 reach all nodes????
```

```
    @reach = @{$conn[0]};         # node 0 can reach all its neighbors
    $reached = $#{$conn[0]}+1;
    $r = 0;
    while($r <= $#reach){
        $ss = $#{$conn[$reach[$r]]};
        for $s (0 .. $ss  ){
        if(memberofreach($conn[$reach[$r]][$s])==0){
            $reach[$reached] = $conn[$reach[$r]][$s];
            $reached++;
            }
        }
        $r++;
    }
    if($reached < $i){  #if node 0 can't reach every other node,
        $connected = 0; #it is not connected
    }else {
        $connected = 1; #otherwise its connected
    }
        ################################################
    print OUTFILE "$input,$connected,$avgdeg\n";
    $avgdeg = 0;
    $i = 0;              #clear count for next set
    }
}
close(OUTFILE);

sub memberofreach {
    my $test = $_[0];
    my $res = 0;
    $w = 0;
    while($w <= $#reach && $res == 0){
        if($reach[$w]==$test){
        $res = 1;
        }
        $w++;
    }
    return $res;
    }
#------------end conntest.pl-------------------------
```

consolemakemfile.pl (makemfile.pl does the same except doesn't write average degree to the command line)

```
#!/usr/local/ActivePerl-5.6/bin/perl -w
#
#makemfile.pl - takes in a .pos file and spits out a file that can be
#opened in Matlab to display a graph of the network
#also outputs to stdout the filename and average degree to be possibly
#read by another script
if(@ARGV>0){
    $filename = $ARGV[0];
    #print $filename."\n";
}else {
    print "give me a filename\n";
    exit(0);
}

if($filename =~ /\/|\\/){ #I have a file in another directory
    @dirs = split /\/|\\/, $filename;
    $numdirs = @dirs;
    $fname = $dirs[$numdirs-1];
}else{
    $fname = $filename;
}
```

```
$l = length($fname);
if($l<23){
    $type = substr($fname, 0, 1);
    $numnodes = substr($fname, 1, 3);
    $seed = int(substr($fname, $l-13, 9));
    if($l == 21){
    $width = substr($fname, 4, 4);
    }else{
    if($l == 22){
        $width = substr($fname, 4, 5);
        }
    }
    if($type == 2 || $type == 6) {
    $ntype = "RU";
    $fulltype = "Random Uniform";
    }else{
    if($type == 4 || $type == 5 || $type == 7){
        $ntype = "CD";
        $fulltype = "Constant Density";
    }else{
        $ntype = "NA";
        $fulltype = "Unknown Type";
        }
    } #create a file name of a standard form
    $mfile = "m".$numnodes.$ntype.$width.$seed.".m";
    $mfile =~ s/\./\_/;
}else{          #longer than expected, just strip out zeros, and set pos extension
    $mfile = $fname;
    $pos = ".pos"; $nada = ""; $zeros = "00000";
    $mfile =~ s/$pos/$nada/;
    $mfile =~ s/\./$nada/;
    $mfile =~ s/$zeros/$nada/;
    $mfile = "m".$mfile.".m";
}

if($filename =~ /\/|\\/){ #I have a file in another directory
    $dirs[$numdirs-1] = $mfile;
    $output = "";
    for($t = 0; $t< $numdirs-1; $t++){
        $output = $output.$dirs[$t]."\\";
        }
    $output = $output.$dirs[$numdirs-1];
    $mfile = $output;
}
%pos = ();                          #a hash of node ids and positions
@deg = ();                          #an array of degrees of each node
open(INFILE, $filename) or die "can't read ".$filename;
@lines = <INFILE>;                                  #read in all lines of file
close(INFILE);
$numnodes = 0;
foreach $line (@lines){                 #loop through all lines
    chop($line);
    chop($line);
    @words = split(',', $line);
    #grab the data
    $name = $words[0];           #first is node id
    $x = $words[1];
    $y = $words[2];
    $namex = $name."x";
    $namey = $name."y";
    $pos{$namex}= $x;
    $pos{$namey}= $y;
    $numnodes++;
}
open(MOUT, ">$mfile");
print MOUT "function createmap()\n \%\% Create figure\n";
```

```
print MOUT "figure1 = figure;\n\%\% Create axes\n";
print MOUT "axes1 = axes('Parent',figure1);\n";
print MOUT "title(axes1,'".$fulltype.", ".$numnodes." Nodes, Width ".$width.
                                                          ", Seed ".$seed."');\n"
print MOUT "xlabel(axes1,'x');\n  ylabel(axes1,'y'); \n";
for($i = 0; $i<$numnodes; $i++)
{
    print MOUT "line([".$pos{$i."x"}.",".$pos{$i."x"}."],";
    print MOUT "[".$pos{$i."y"}.",".$pos{$i."y"}."],";
    print MOUT " 'Color', 'b', 'Marker', 'o', 'MarkerSize', 5, 'MarkerFaceColor', ";
    print MOUT "'r')\n";
    $d=0;
    for($j=0; $j<$numnodes; $j++)
    {
    $dist = sqrt(($pos{$j."x"}-$pos{$i."x"})**2+($pos{$j."y"}-$pos{$i."y"})**2 );
    if($dist <= 15 && $i!=$j)
        {
        print MOUT "line([".$pos{$i."x"}.",".$pos{$j."x"}."],[".$pos{$i."y"}.",";
        print MOUT "$pos{$j."y"}."], 'Color', 'b', 'Marker', 'o', 'MarkerSize', 5,";
        print MOUT " 'MarkerFaceColor', 'r')\n";
        $d++;
        }
    }
    $deg[$i] = $d;
}
close(MOUT);
$totdeg = 0;
for($i = 0; $i < $numnodes; $i++)
{
    $totdeg = $totdeg + $deg[$i];
}
$avgdeg = $totdeg/$numnodes;
print "$filename,$avgdeg,\n";
#------------end consolemakemfile.pl-------------------------
```

Appendix E. Best, Worst and Average ADE Performance Graphs

This appendix contains graphs of the networks on which AFL and Map Growing had their best, worst and average ADE performance. For each network, the original graph and the resulting output from AFL and Map Growing are presented. Since both algorithms can produce networks in a rotated and flipped coordinate system from the original, the graphs resulting from the algorithms have been rotated and flipped by hand to best approximate the original coordinate system for comparison. In each of the resulting graphs, white nodes indicate the node 0 and the five reference nodes for AFL or the starting triangle for Map Growing.

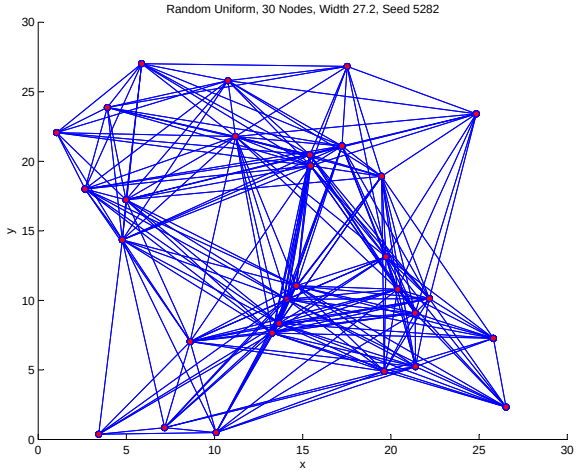


Figure E.1: Graph of Best AFL Network

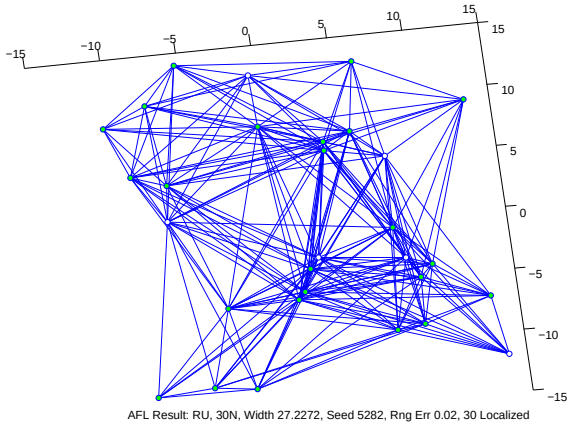


Figure E.2: AFL results, ADE=0.072

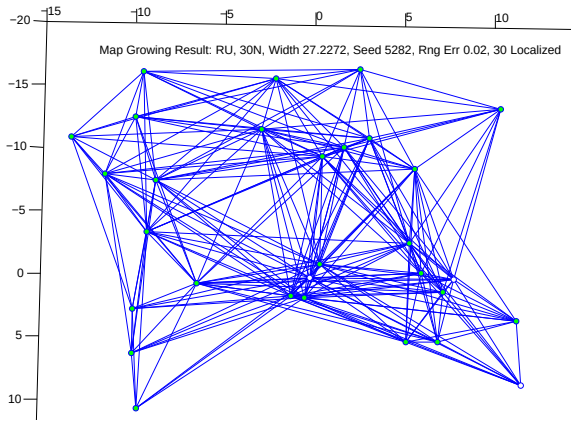


Figure E.3: Map Growing results, ADE=0.795

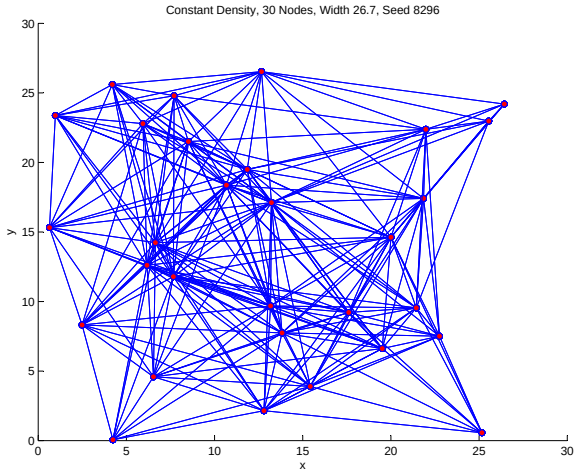


Figure E.4: Graph of Best Map Growing Network

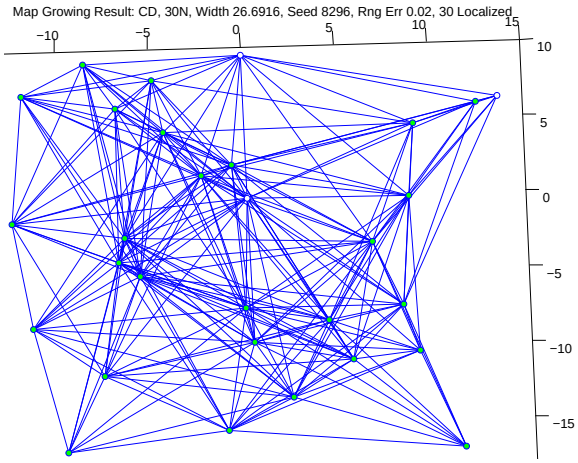


Figure E.5: Map Growing results, ADE=0.26

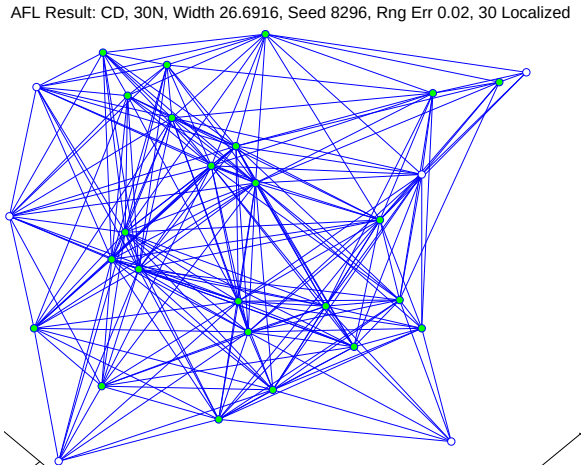


Figure E.6: AFL results, ADE=0.11

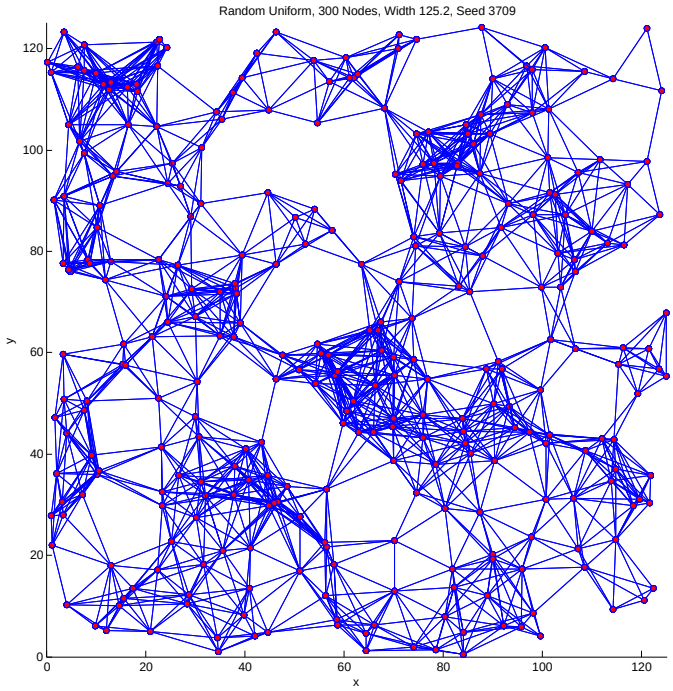


Figure E.7: Average AFL Graph

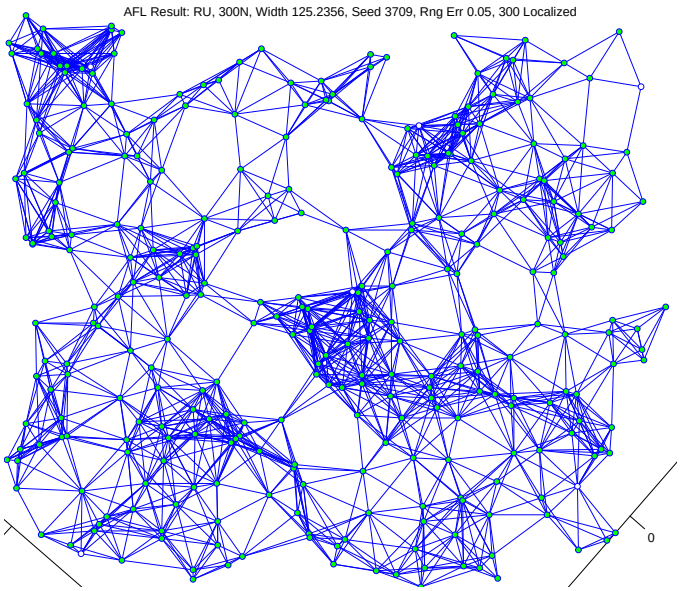


Figure E.8: AFL results, ADE=3.36

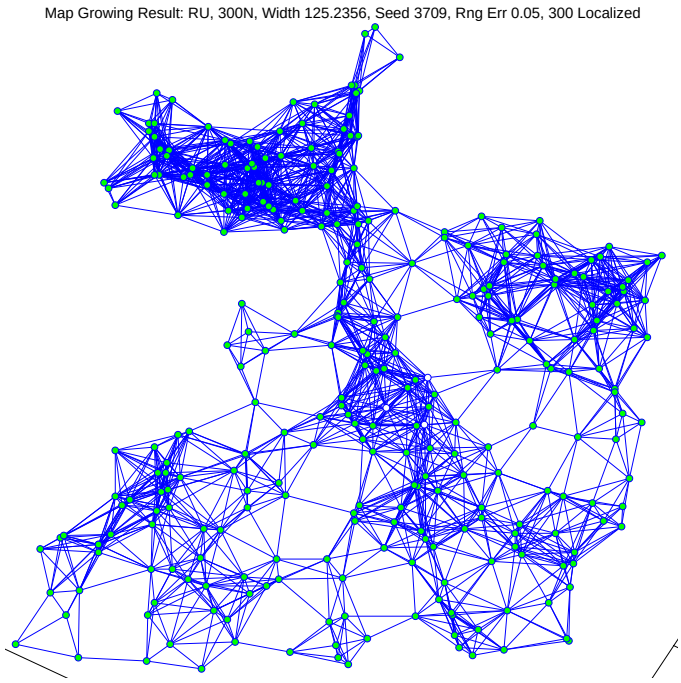


Figure E.9: Map Growing results, ADE=8.62

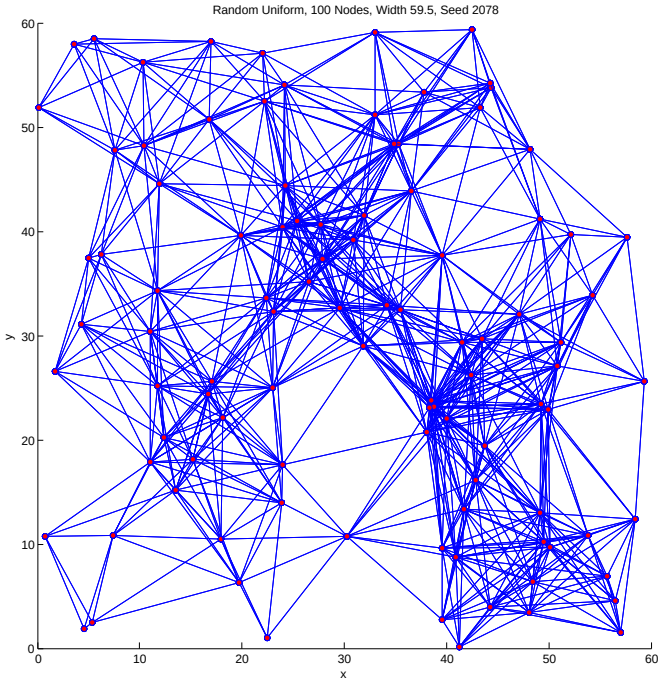


Figure E.10: Average Map Growing Graph

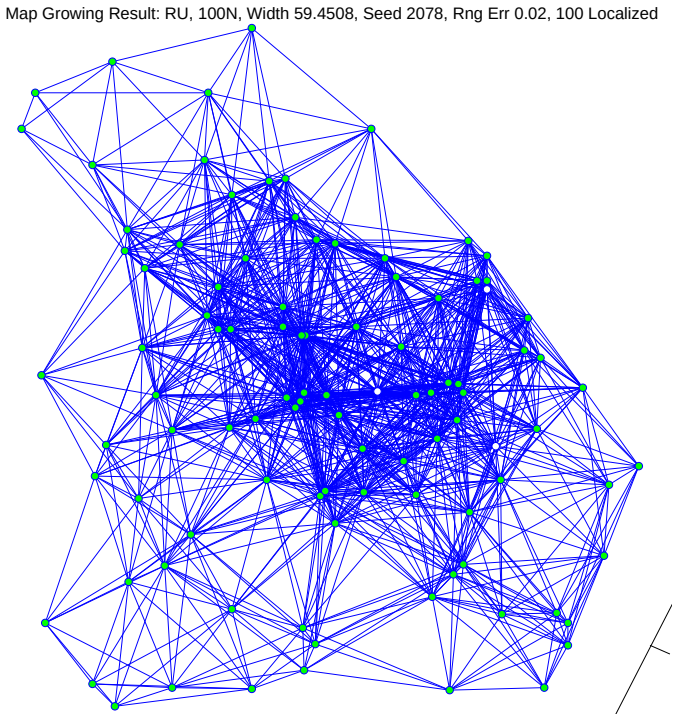


Figure E.11: Map Growing results, ADE=6.81

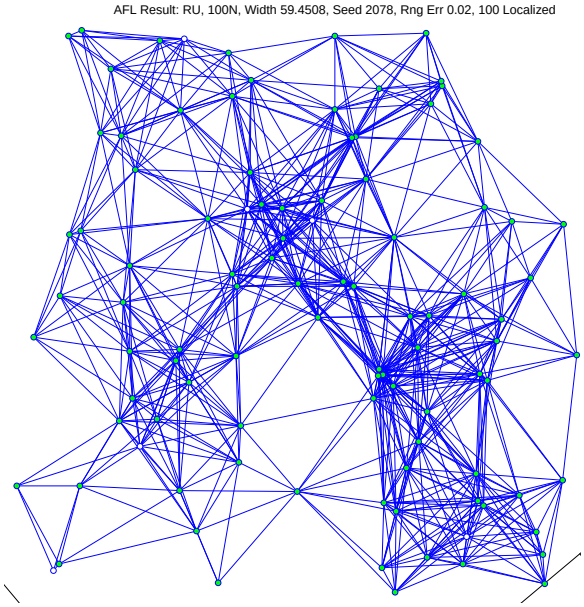


Figure E.12: AFL results, ADE=0.33

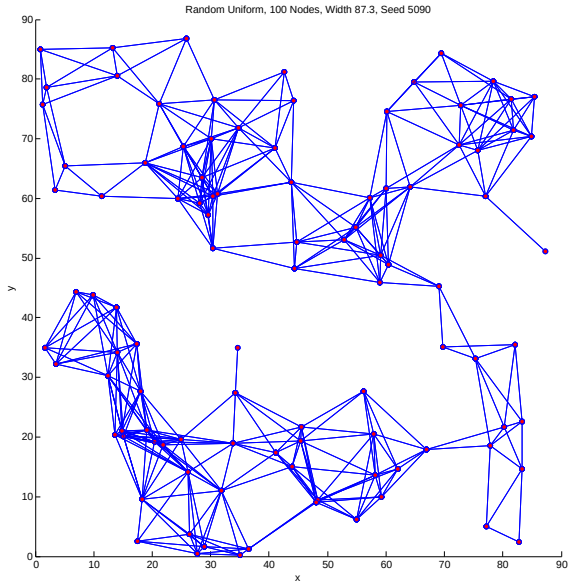


Figure E.13: Worst AFL Graph

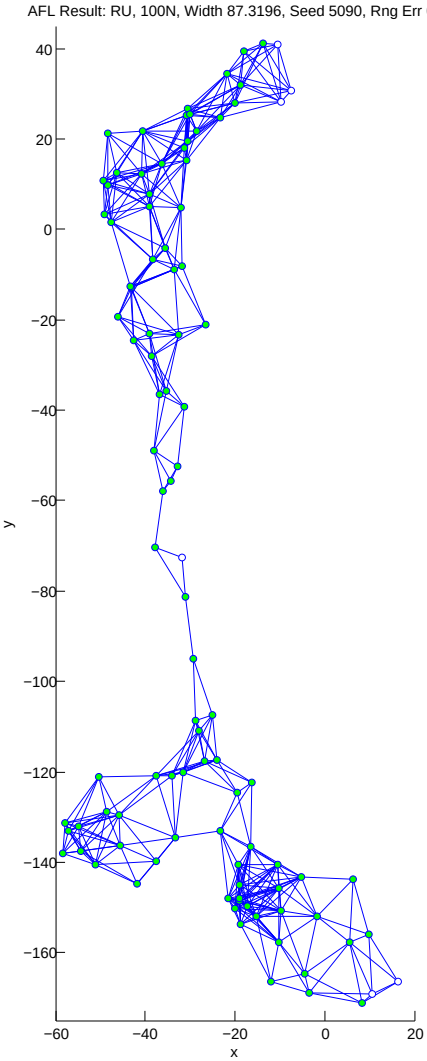


Figure E.14: AFL results, ADE=45.5

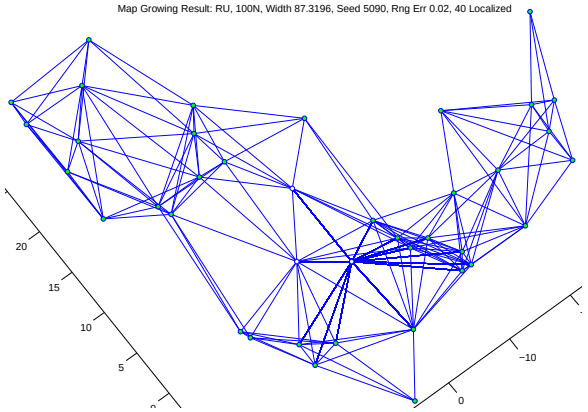


Figure E.15: Map Growing results, ADE=1.35

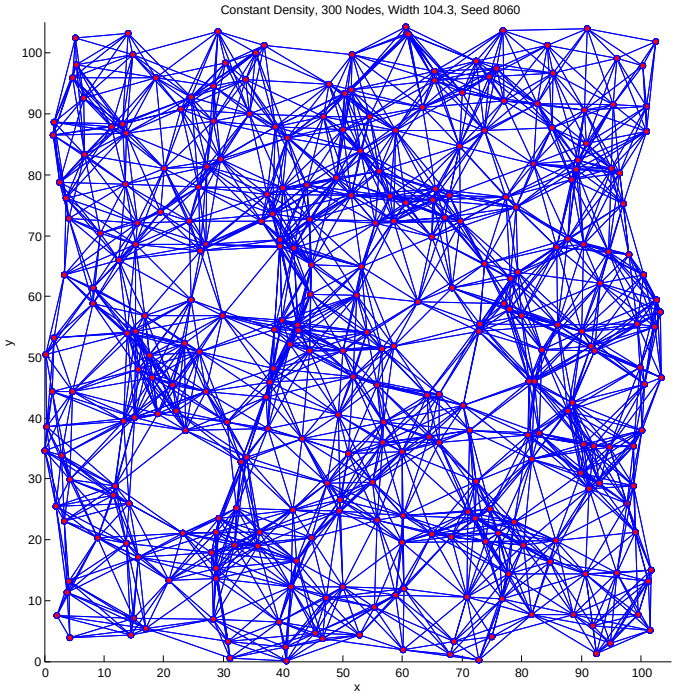


Figure E.16: Worst Map Growing Graph

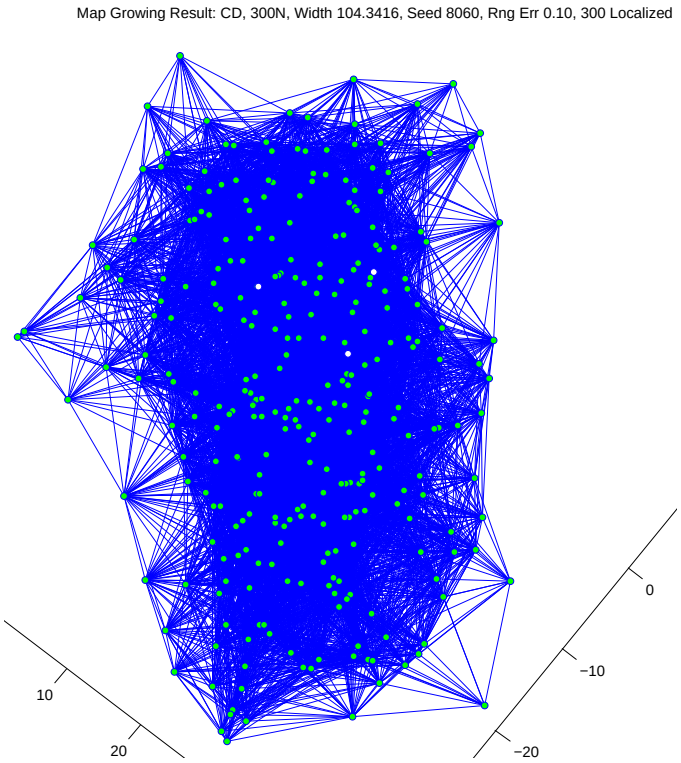


Figure E.17: Map Growing results, ADE=32.0

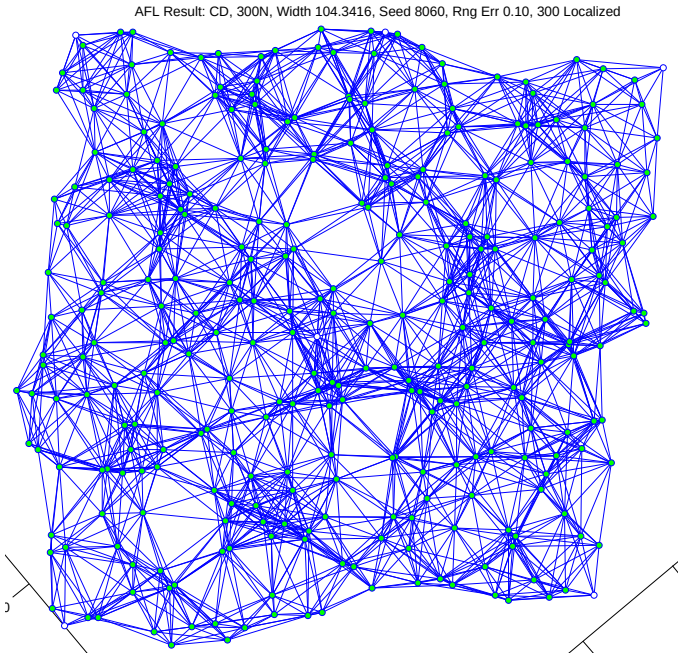


Figure E.18: AFL results, ADE=2.53

Appendix F. Verification of ANOVA Assumptions

This appendix includes the figures that verify the ANOVA assumptions. Also, additional figures are included that illustrate why some data points are removed from the study.

Map Growing

Upon initial inspection of the normal probability plot and residuals versus fitted values plot, the Map Growing Average Transmitted Bits exhibited a high number of low outliers that were linear but at a different slope than the normal distribution line. This is shown in Figures F.1 and F.2. This suggested that there were at least two modes of operation within the data. Examining the low outliers revealed that most of the data points corresponded to Random Uniform, degree 8, larger sized networks. The RU, degree 8 networks with 100 or 300 nodes also have the lowest percent localized results. Figure H.10 in Appendix H shows that the only group of networks that localize less than 90% with Map Growing are RU, degree 8, with 100 or 300 nodes. The average percent localized for this class of networks is 65.85%. All other networks localize 98.54% using Map Growing.

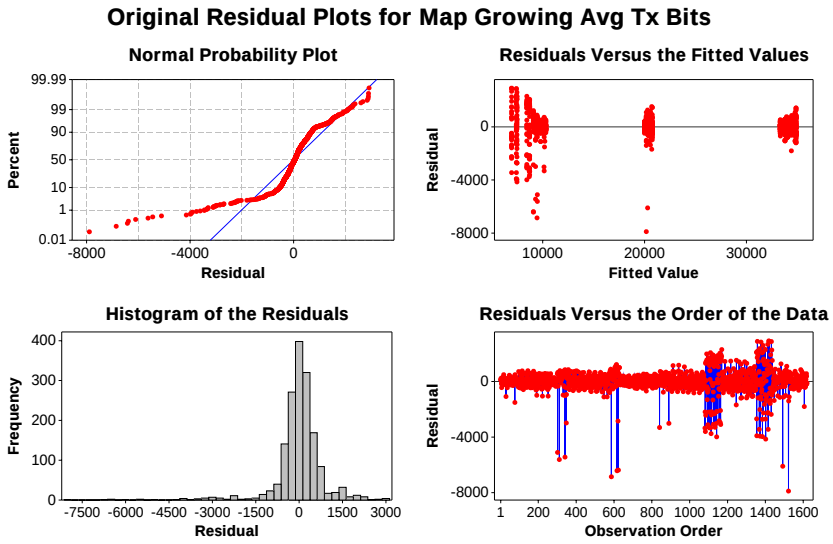


Figure F.1: Original Residual Plots of Map Grow Average Transmitted Bits

Since RU networks have areas of high degree nodes and areas with low degree nodes, they are more likely than the CD networks to not localize all nodes, when using Map Growing. The low degree nodes in RU networks have more trouble finding eligible nodes to localize with (noncollinear with acceptable residual, or having localized neighbors that can be used with the 2 Beacon Solver). The lower the average degree of the entire network, the worse this effect becomes. The phenomenon worsens as the network size grows. With only 30 nodes, it is more likely the map can grow to all nodes. With 300 nodes, there is more of a chance that the map growth will not reach more nodes because of areas of low degree. As a result, Map Growing with RU, degree 8 networks with 100 and 300 nodes localizes a low percentage of nodes. Since fewer nodes localize, fewer nodes are broadcasting their positions, and their localized neighbors. Overall, messages decrease as well as transmitted bits. This effect causes a second mode of operation - one in which few nodes localize (<75%).

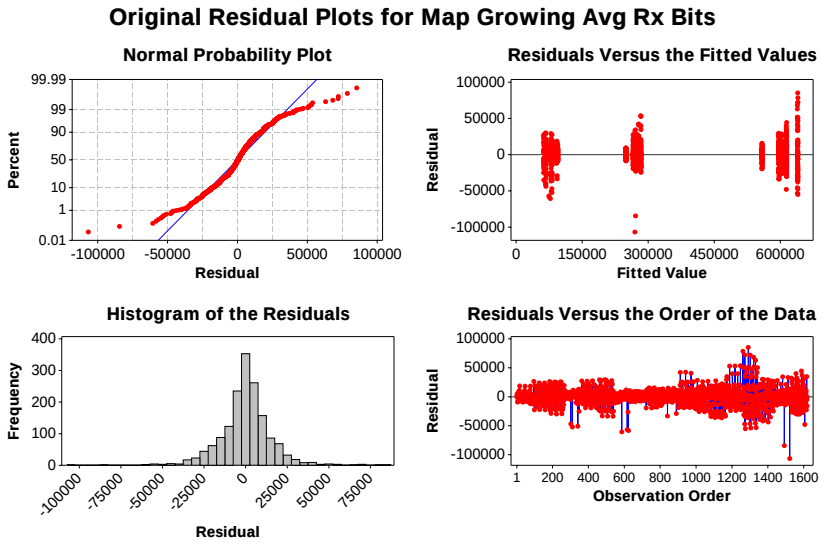


Figure F.2: Original Residual Plots of Map Grow Average Received Bits

To produce a valid ANOVA, the degree 8 networks were partitioned from the data set. Additional, outliers were also removed, data points corresponding to a high number of received bits. Examining the histogram of degrees of the nodes in the network and comparing it to a typical performing network shows why this outlier

network received an usually high number of bits. Figure F.3 is a histogram of the degrees from the outlier network. Figure F.4 is a histogram of the degrees from a typical network. In the outlier network, a disproportionately high number of nodes have a very high degree offset by nodes that have a very low degree. In the typical case, the degrees are more normally distributed with most nodes having degrees close to or at 16 and less nodes having degrees far from 16. Since the outlier network had an usually high number of nodes with a very high degree, most of the messages sent were heard by more than the typical number of nodes. This effect drives up the overall number of bits heard. Since the goal is to characterize typical behavior, this seed was removed from every set. As such, the Map Growing ANOVAs do not reflect behavior that includes networks with nodes that have a distribution of degrees favoring high degrees.

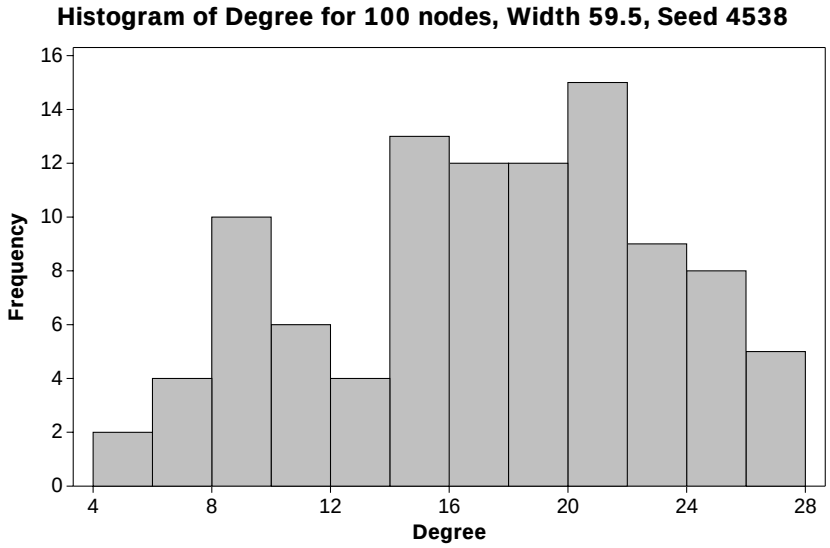


Figure F.3: Histogram of Node Degree - Outlier Case, 100 Nodes, Random Uniform, Avg Degree 16

Lastly, a set of data points were observed to have a very low number of transmitted bits. Examining the degrees of the three starting nodes reveals that the outlier data points all had sets of starting nodes with unusually high degrees. The starting nodes are the first set of 3 that localize in Map Growing. Once they localize, they

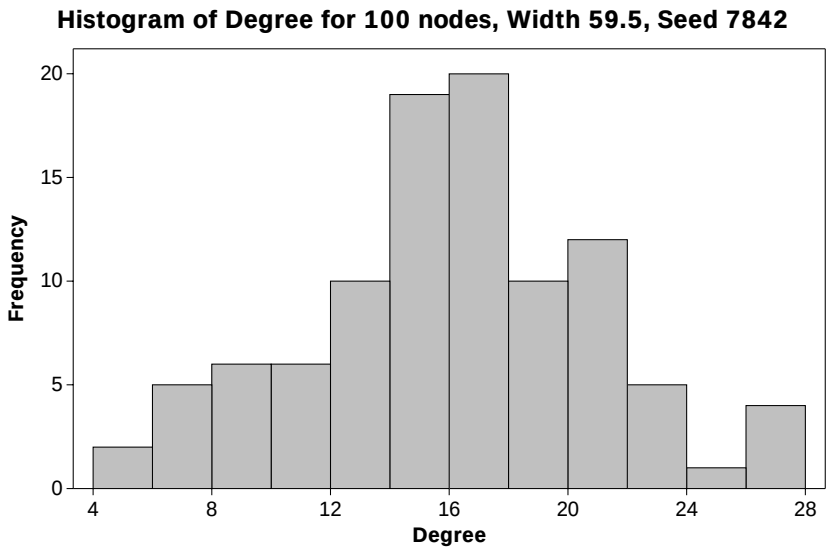


Figure F.4: Histogram of Node Degree - Typical Case, 100 Nodes, Random Uniform, Avg Degree 16

broadcast their information to their neighbors. If a node hears all three broadcasts and the starting nodes are not collinear, it can immediately localize. If the starting nodes have a very high degree, it is more likely that more nodes can instantly localize, eliminating the need to transmit any more bits. This results in a very easy graph for Map Growing to localize, and not the typical case. The removal of these points and others to keep the experiment balanced leaves 23 repetitions for each configuration in the Map Growing data set.

The Normal Probability Plot of the Map Growing Average Received Bits and Average Transmitted Bits are shown in Figures F.5 and F.7. These plots show that the residuals approximately follow the normal distribution line and confirms that the residuals in the measurements are normally distributed. To confirm errors are independent of the factor levels, a plot of the Residuals vs Fitted Values is observed to confirm no trends are present. The same plot can be used to verify that the errors have the same variance for all factor levels by confirming a constant height in the spread of residuals. As can be seen in Figure F.6 and F.8, there are no trends in the residuals versus fitted data for either average bits transmitted or average bits received

and the height of the residual plots are generally consistent. Additionally, since the magnitude of the residuals are less than an order of magnitude than the fitted value, any trends can be ignored [Jai91]. The errors are independent and the variance is constant.

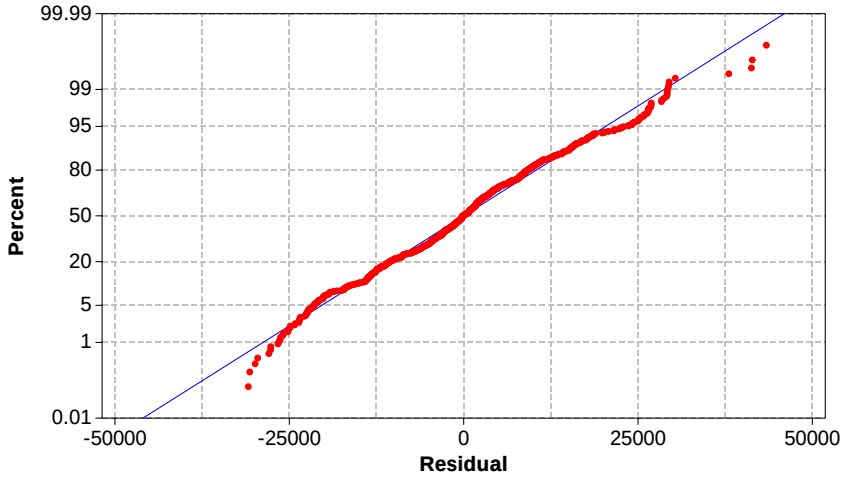


Figure F.5: Map Growing: Normal Probability Plot of Map Growing Average Received Bits

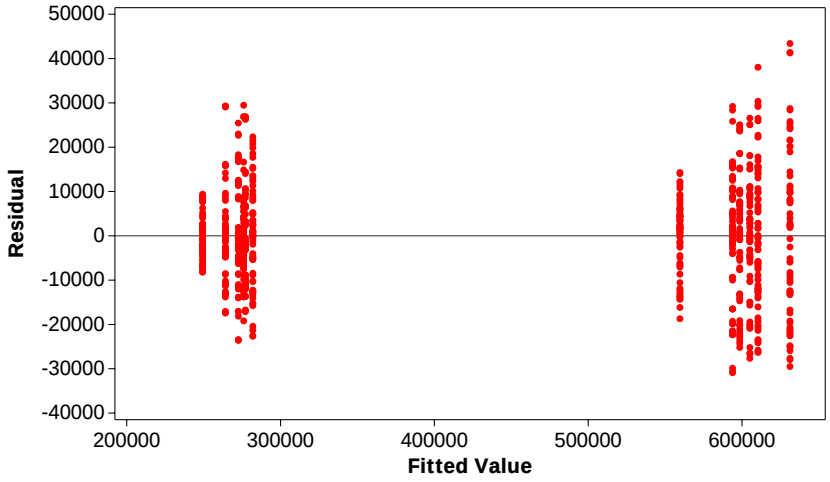


Figure F.6: Map Growing: Residuals vs Fitted Values of Map Growing Average Received Bits

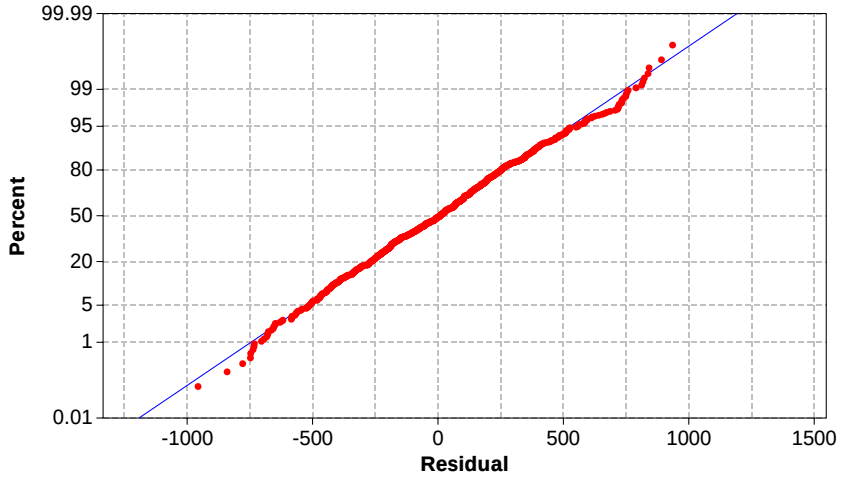


Figure F.7: Map Growing: Normal Probability of Plot of Map Growing Average Transmitted Bits

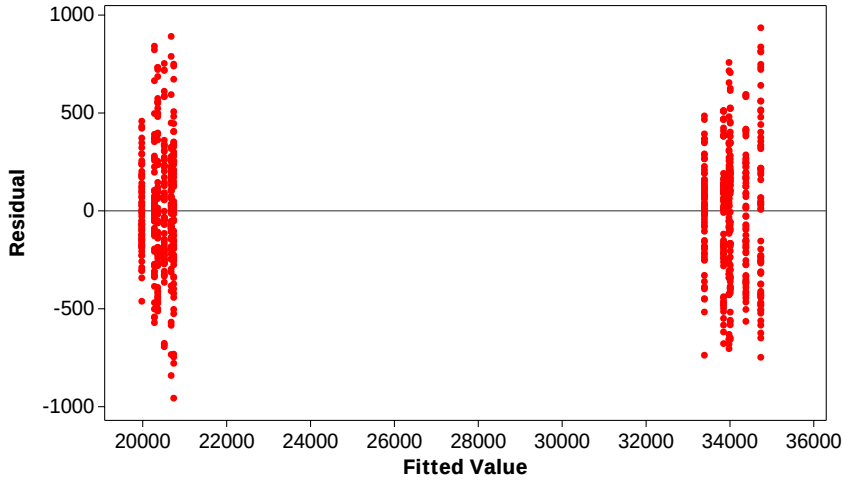


Figure F.8: Map Growing: Residuals vs Fitted Values of Map Growing Average Transmitted Bits

AFL

The original residual versus fitted value graphs indicates an increase in variance of the errors as degree increased. This suggests a system that is not additive and requires a transformation in the data [Jai91]. The transformation that reduced the variance in the errors, trends in the variance and resulted in errors that are normally distributed is a natural log transformation. As a result, the AFL ANOVAs operate on the natural log of the average received bits and average transmitted bits.

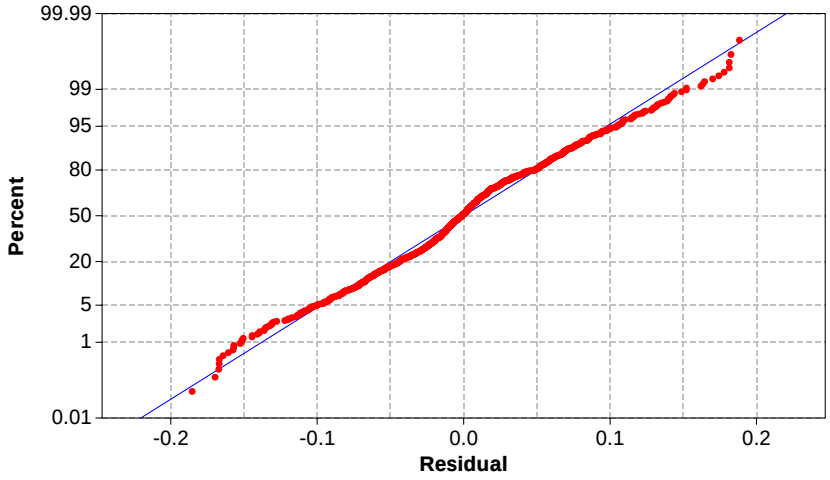


Figure F.9: AFL: Normal Probability Plot of ln(ARB)

Initial residual plots for AFL communication data also had a series of outliers at both ends for both transmitted and received bits. In general, nodes can require more iterations if they happen have neighbors with high errors in opposite directions. This would create a set of position forces that would require many iterations for the node to balance. This is made worse if a node has a low degree. With a low degree, there are fewer sources from which a position correction can be provided. Similarly, a node with a given number of neighbors is better served by having those neighbors spread around the node evenly, rather on one side only. This creates more directions from which the position corrections are provided. In the case of the outliers, most are networks that have many areas that are concave, or that curves inward. With many concave areas, there are more nodes that seem like the are on the edge of the network with most of their neighbors on one side. For these nodes, there are less directions to help balance the position forces, causing them to require more iterations, and more transmitted and received bits. Most of the low AFL outliers were found to be networks that are very balanced in terms of the degrees of the nodes and are mostly convex, with no or few curves inward from the edge of the network. This represented the other extreme from the concave networks. In order to characterize the typical performance both sets

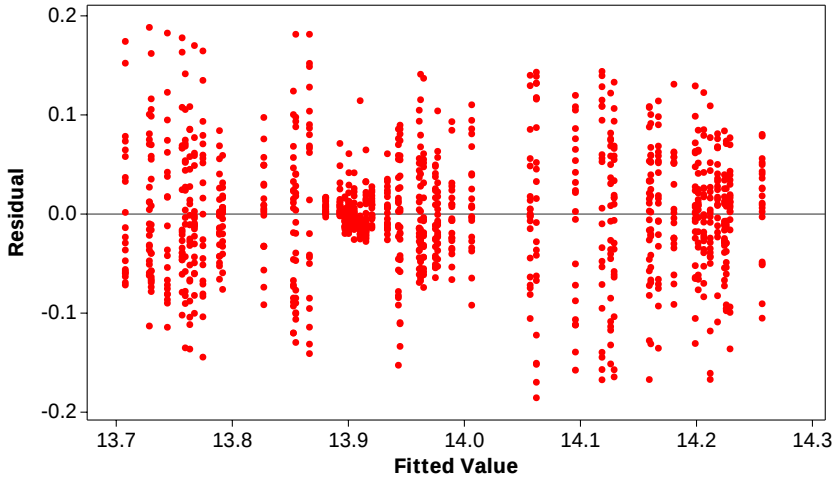


Figure F.10: AFL: Residuals vs Fitted Values of ln(ARB)

were removed as well as repetitions from other configurations to maintain a balanced experiment. This leaves 21 repetitions of each configuration.

The final residual plots for the AFL communication data is in Figures F.9 to F.12. The AFL normal probability plots for Natural Log of Average Transmitted Bits (ln(ATB)) and Average Received Bits (ln(ARB)), in Figures F.9 and F.11, are both approximately linear confirming that these errors are also normally distributed. The Residuals versus Fitted Values figures for both responses confirm that the errors are independent with constant variance. The magnitude of the errors are an order of magnitude less than the fitted values, allowing any trends, if they existed, to be ignored.

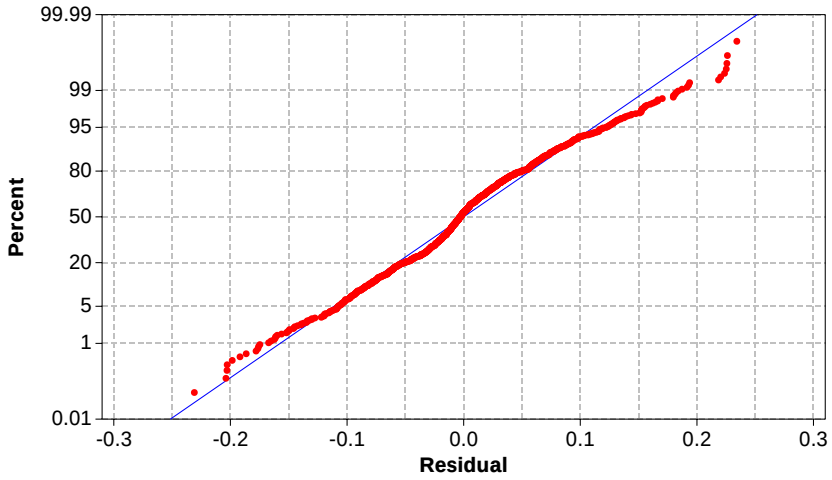


Figure F.11: AFL: Normal Probability Plot of ln(ATB)

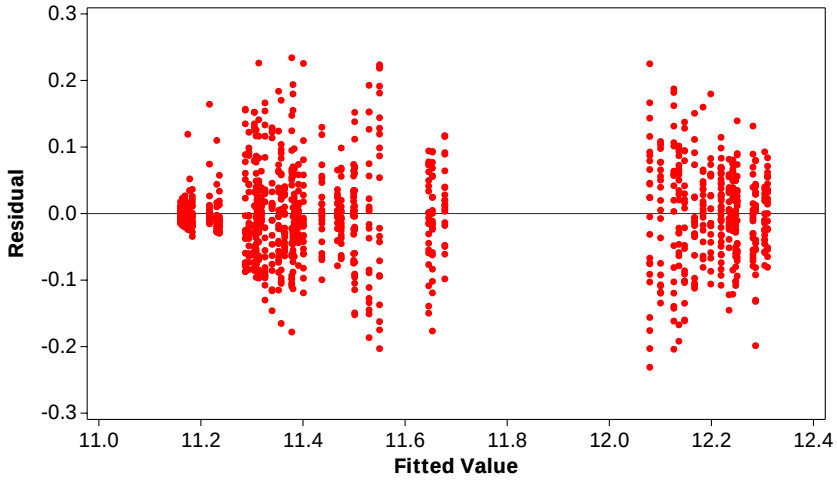


Figure F.12: AFL: Residuals vs Fitted Values of ln(ATB)

Appendix G. Concave and Convex Networks

This appendix contains examples of concave and convex networks.

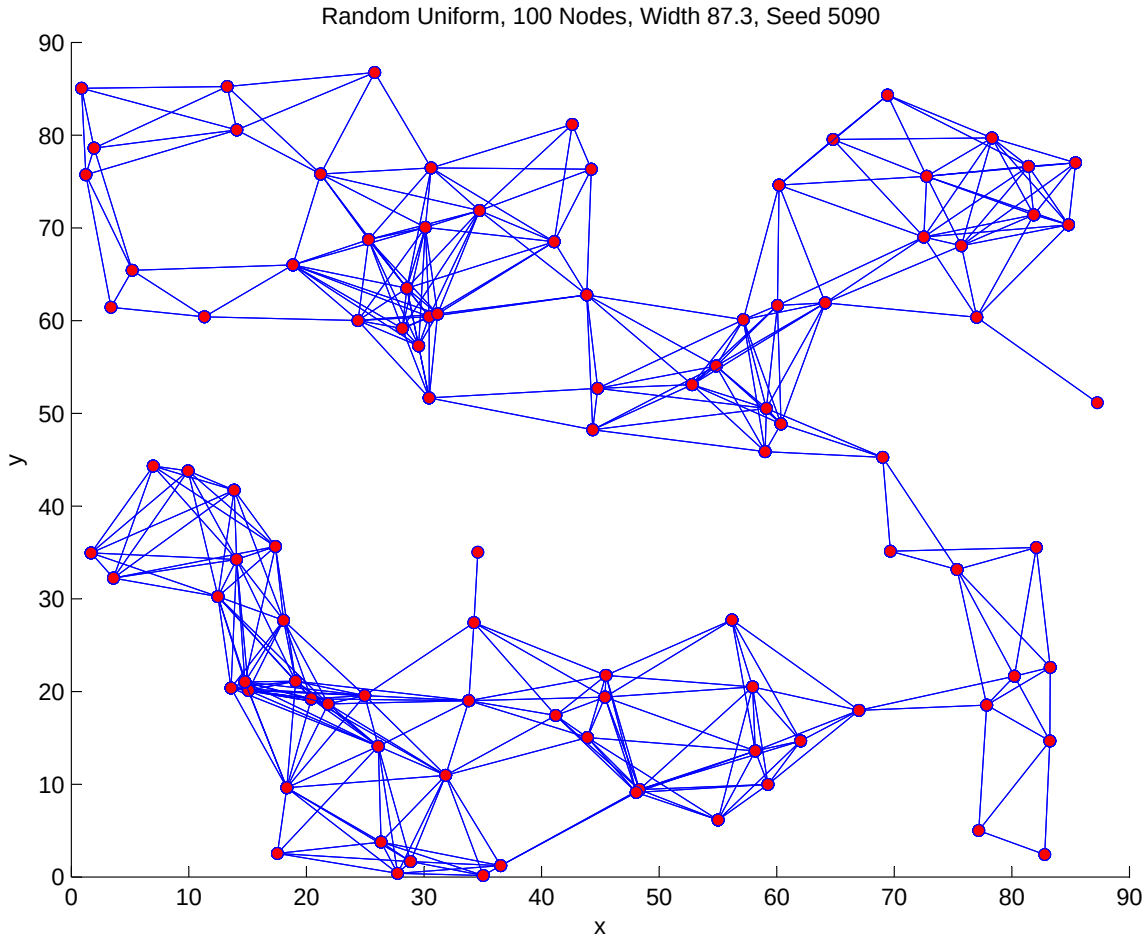


Figure G.1: Concave Example: 100 Nodes, Random Uniform, Degree 8, ADE = 43.5

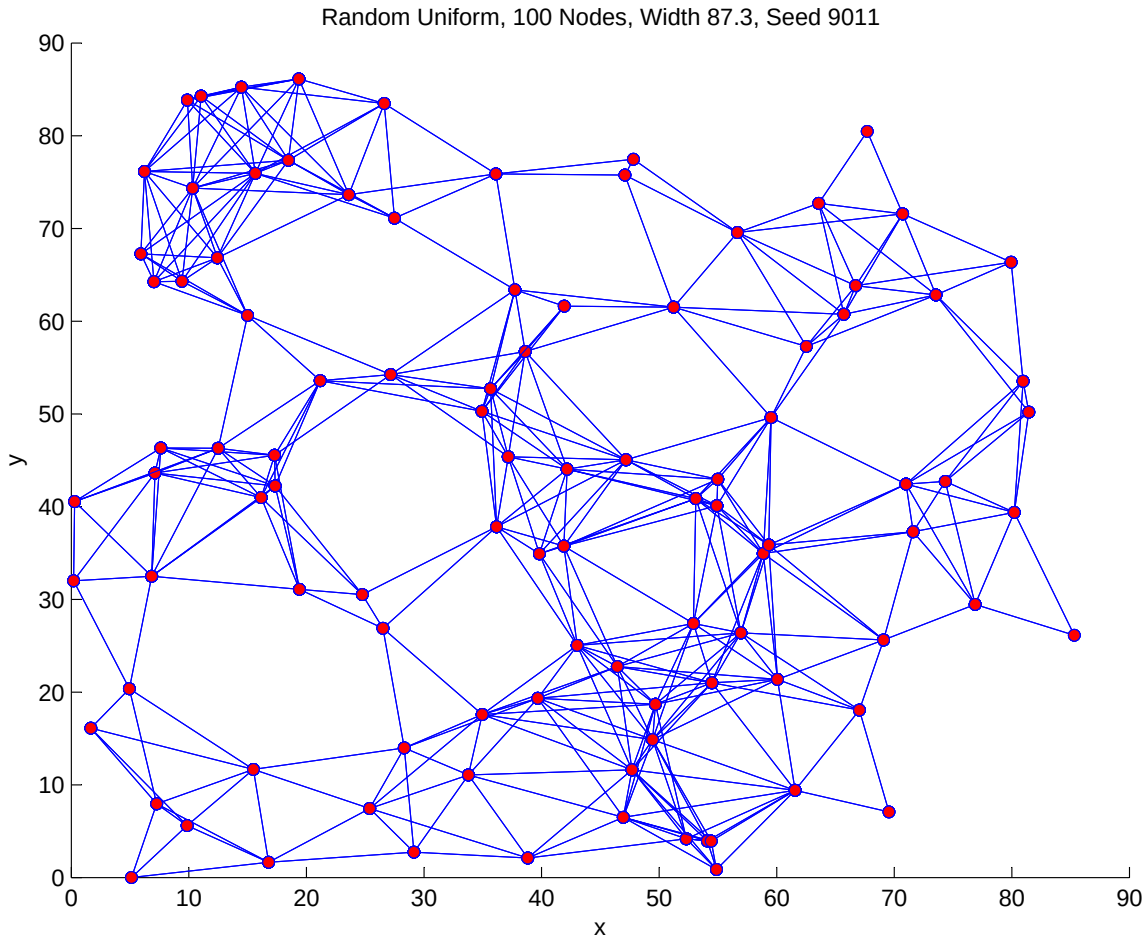


Figure G.2: Convex Example: 100 Nodes, Random Uniform, Degree 8, ADE = 1.1

(For all computations and tables , $\alpha = 0.1$, RE = Range Error, Col = Column.)

Map Growing

Table H.1: Computation of Effects for Map Growing Average Transmitted Bits

		CD			RU					
		Degree	Degree	Degree	Degree	Degree	Degree	Row	Row	Row
Size	RE	8	12	16	8	12	16	Sum	Mean	Effect
30	0.02	10216.23	20325.38	33778.88	10312.49	20557.14	33946.62	3874102.40	21522.79	297.75
	0.05	10243.71	20309.55	33777.02	10152.42	20561.92	33997.74	3871270.90	21507.06	282.02
	0.10	10178.55	20339.17	33793.70	10132.24	20553.60	33961.20	3868754.23	21493.08	268.04
100	0.02	9962.86	20358.35	33996.65	8714.71	20765.61	34869.07	3860017.87	21444.54	219.50
	0.05	9477.03	20379.65	34007.19	8681.73	20753.71	34871.91	3845136.87	21361.87	136.83
	0.10	9300.37	20329.12	34079.64	8439.60	20707.08	34778.23	3829021.42	21272.34	47.30
300	0.02	9796.87	20035.53	33328.30	7417.04	20579.54	34421.62	3767367.98	20929.82	-295.23
	0.05	9447.61	19996.78	33337.30	7468.64	20300.64	34445.90	3749905.86	20832.81	-392.23
	0.10	9088.53	19990.98	33357.29	6945.28	20175.70	34408.69	3718994.01	20661.08	-563.97
Col Sum		2631353.17	5461935.54	9103679.31	2347924.34	5548648.45	9291029.73	34384570.55		
Col Mean		9745.75	20229.39	33717.33	8696.02	20550.55	34411.22		**21225.04**	
Col Effect		-11479.29	-995.65	12492.29	-12529.03	-674.49	13186.18			

Table H.2: Table of Effects for Map Growing Average Transmitted Bits

Parameter	Mean Effect	Std Dev	Confidence Interval	
Mean	21225.04	21.83	21189.13	21260.96
Range Error Effects				
0.02	74.01	30.88	23.21	124.80
0.05	8.87	30.88	-41.92	59.67
0.10	-82.88	30.88	-133.67	-32.08
Degree Effects				
8.00	-12004.16	30.88	-12054.95	-11953.36
12.00	-835.07	30.88	-885.87	-784.28
16.00	12839.23	30.88	12788.44	12890.03
Size Effects				
30.00	282.60	30.88	231.81	333.39
100.00	134.54	30.88	83.75	185.34
300.00	-417.14	30.88	-467.94	-366.35
Type Effects				
CD	5.78	21.83	-30.14	41.70
RU	-5.78	21.83	-41.70	30.14

Table H.3: Contrast Results for Map Growing Average Transmitted Bits

Parameter	Level_A	Level_B	Mean Difference	Lower CL	Upper CL
	0.02	0.05	65.14	14.34	115.93
RE	0.02	0.1	91.75	106.09	207.68
	0.05	0.1	156.88	40.95	142.54
	8	12	-11169.09	-11219.88	-11118.29
Degree	8	16	-24843.39	-24894.19	-24792.60
	12	16	-13674.31	-13725.10	-13623.51
	30	100	148.06	97.26	198.85
Size	30	300	699.74	648.95	750.54
	100	300	551.68	500.89	602.48
Type	CD	RU	11.56	-39.23	62.36

Table H.4: Computation of Effects for Map Growing Average Received Bits

		CD			RU					
Size	RE	Degree 8	Degree 12	Degree 16	Degree 8	Degree 12	Degree 16	Row Sum	Row Mean	Row Effect
30	0.02	92342.39	271562.20	597316.93	94694.58	278208.64	603247.39	58121163.97	322895.36	4959.56
	0.05	92473.05	271256.52	597378.73	93424.35	278255.19	604352.99	58114224.80	322856.80	4921.01
	0.10	92059.30	271713.22	597654.43	93250.52	278204.95	603472.24	58090640.13	322725.78	4789.99
100	0.02	85503.03	265968.51	596308.01	81694.88	283508.99	639740.57	58581719.89	325454.00	7518.21
	0.05	81363.50	266358.54	596478.56	81268.06	283451.98	639431.22	58450555.92	324725.31	6789.52
	0.10	79988.63	265869.06	597588.99	79394.22	283001.76	638560.47	58332094.17	324067.19	6131.40
300	0.02	80975.60	249637.62	558135.38	67174.66	274897.24	613592.66	55332394.58	307402.19	-10533.60
	0.05	77854.70	249131.67	558426.17	67874.19	271077.74	614268.44	55158987.26	306438.82	-11496.97
	0.10	74836.38	249158.76	558846.85	62956.41	269733.59	613608.08	54874202.40	304856.68	-13079.11
Col Sum		22721897.35	70819683.26	157744021.71	21651955.87	75010202.88	167108222.06	515055983.12		
Col Mean		84155.18	262295.12	584237.12	80192.43	277815.57	618919.34		**317935.79**	
Col Effect		-233780.62	-55640.67	266301.33	-237743.36	-40120.23	300983.55			

Table H.5: Table of Effects for Map Growing Average Received Bits

Parameter	Mean Effect	Std Dev	Confidence Interval	
Mean	317935.79	385.30	317301.97	318569.61
Range Error Effects				
0.02	648.06	544.90	-248.30	1544.41
0.05	71.19	544.90	-825.17	967.54
0.10	-719.24	544.90	-1615.60	177.11
Degree Effects				
8	-235761.99	544.90	-236658.34	-234865.64
12	-47880.45	544.90	-48776.80	-46984.09
16	283642.44	544.90	282746.08	284538.79
Size Effects				
30	4890.19	544.90	3993.83	5786.54
100	6813.04	544.90	5916.69	7709.39
300	-11703.23	544.90	-12599.58	-10806.88
Type Effects				
CD	-7706.65	385.30	-8340.47	-7072.84
RU	7706.65	385.30	7072.84	8340.47

Table H.6: Contrast Results for Map Growing Average Received Bits

Parameter	Level_A	Level_B	Mean Difference	Lower CL	Upper CL
	0.02	0.05	576.87	-319.48	1473.22
RE	0.02	0.1	790.43	470.95	2263.65
	0.05	0.1	1367.30	-105.93	1686.78
	8	12	-187881.54	-188777.90	-186985.19
Degree	8	16	-519404.43	-520300.78	-518508.07
	12	16	-331522.88	-332419.24	-330626.53
	30	100	-1922.85	-2819.21	-1026.50
Size	30	300	16593.42	15697.06	17489.77
	100	300	18516.27	17619.92	19412.62
Type	CD	RU	-15413.31	-16309.66	-14516.95

Table H.7: Computation of Effects for Map Growing Average Distance Error

		CD			RU					
		Degree	**Degree**	**Degree**	**Degree**	**Degree**	**Degree**	**Row**	**Row**	**Row**
Size	**RE**	**8**	**12**	**16**	**8**	**12**	**16**	**Sum**	**Mean**	**Effect**
	0.02	0.85	0.72	0.54	1.01	0.72	0.58	132.67	0.74	-6.05
30	**0.05**	1.74	1.47	1.11	2.29	1.43	1.29	279.89	1.55	-5.24
	0.10	2.93	2.68	2.10	3.03	2.31	2.15	456.49	2.54	-4.26
	0.02	3.09	1.78	2.02	2.92	2.86	2.04	441.25	2.45	-4.34
100	**0.05**	5.15	3.87	3.80	4.77	4.85	3.89	789.97	4.39	-2.40
	0.10	8.02	7.63	7.69	6.56	9.05	8.98	1437.98	7.99	1.20
	0.02	8.90	5.82	8.29	8.10	8.16	8.20	1423.96	7.91	1.12
300	**0.05**	14.02	11.46	14.25	11.82	13.09	13.73	2351.10	13.06	6.27
	0.10	19.69	19.47	22.30	16.09	22.50	22.88	3687.95	20.49	13.70
Col Sum		1931.86	1647.54	1862.95	1697.42	1949.16	1912.33	11001.27		
Col Mean		7.16	6.10	6.90	6.29	7.22	7.08		**6.79**	
Col Effect		0.36	-0.69	0.11	-0.50	0.43	0.29			

Table H.8: Table of Effects for Map Growing Average Distance Error

Parameter	**Mean Effect**	**Std Dev**	**Confidence Interval**	
Mean	6.79	0.08	6.65	6.93
Range Error Effects				
0.02	-3.09	0.12	-3.29	-2.89
0.05	-0.46	0.12	-0.65	-0.26
0.10	3.55	0.12	3.35	3.74
Degree Effects				
8	-0.07	0.12	-0.27	0.13
12	-0.13	0.12	-0.33	0.07
16	0.20	0.12	0.00	0.40
Size Effects				
30	-5.18	0.12	-5.38	-4.98
100	-1.85	0.12	-2.04	-1.65
300	7.03	0.12	6.83	7.23
Type Effects				
CD	-0.07	0.08	-0.21	0.07
RU	0.07	0.08	-0.07	0.21

Table H.9: Contrast Results for Map Growing Average Distance Error

Parameter	Level_A	Level_B	Mean Difference	Lower CL	Upper CL
	0.02	0.05	-2.64	-2.83	-2.44
RE	0.02	0.1	-4.00	-6.83	-6.44
	0.05	0.1	-6.64	-4.20	-3.81
	8	12	0.06	-0.14	0.26
Degree	8	16	-0.27	-0.47	-0.07
	12	16	-0.33	-0.53	-0.13
	30	100	-3.33	-3.53	-3.14
Size	30	300	-12.21	-12.41	-12.01
	100	300	-8.88	-9.07	-8.68
Type	CD	RU	-0.14	-0.34	0.05

Table H.10: Computation of Effects for Map Growing Percent Localized

		CD			RU					
		Degree	**Degree**	**Degree**	**Degree**	**Degree**	**Degree**	**Row**	**Row**	**Row**
Size	**RE**	**8**	**12**	**16**	**8**	**12**	**16**	**Sum**	**Mean**	**Effect**
	0.02	96.56	100.00	100.00	96.22	99.89	100.00	17780.00	98.78	3.87
30	**0.05**	97.67	100.00	100.00	95.33	99.89	100.00	17786.67	98.81	3.91
	0.10	96.67	100.00	100.00	94.89	99.78	100.00	17740.00	98.56	3.65
	0.02	96.77	99.47	99.90	72.03	98.33	99.80	16989.00	94.38	-0.52
100	**0.05**	93.87	99.57	99.87	73.37	98.50	99.83	16950.00	94.17	-0.74
	0.10	92.27	99.23	99.97	69.10	98.27	99.77	16758.00	93.10	-1.80
	0.02	95.11	99.88	100.00	61.11	98.98	99.81	16646.67	92.48	-2.42
300	**0.05**	94.39	99.86	100.00	62.44	98.46	99.84	16649.67	92.50	-2.41
	0.10	93.49	99.79	100.00	57.02	97.93	99.88	16443.33	91.35	-3.55
Col Sum		25703.33	26933.67	26992.00	20445.67	26700.67	26968.00	153743.33		
Col Mean		95.20	99.75	99.97	75.72	98.89	99.88		**94.90**	
Col Effect		0.29	4.85	5.07	-19.18	3.99	4.98			

Table H.11: Table of Effects for Map Growing Percent Localized

Parameter	Mean Effect	Std Dev	Confidence Interval	
Mean	94.90	0.21	94.56	95.25
Range Error Effects				
0.02	0.31	0.30	-0.18	0.80
0.05	0.26	0.30	-0.23	0.75
0.10	-0.57	0.30	-1.06	-0.08
Degree Effects				
8	-9.44	0.30	-9.93	-8.95
12	4.42	0.30	3.93	4.91
16	5.02	0.30	4.53	5.51
Size Effects				
30	3.81	0.30	3.32	4.30
100	-1.02	0.30	-1.51	-0.53
300	-2.79	0.30	-3.28	-2.30
Type Effects				
CD	3.40	0.21	3.06	3.75
RU	-3.40	0.21	-3.75	-3.06

Table H.12: Contrast Results for Map Growing Percent Localized

Parameter	Level_A	Level_B	Mean Difference	Lower CL	Upper CL
	0.02	0.05	0.05	-0.44	0.55
RE	0.02	0.1	0.82	0.39	1.37
	0.05	0.1	0.88	0.33	1.32
	8	12	-13.86	-14.35	-13.37
Degree	8	16	-14.46	-14.96	-13.97
	12	16	-0.60	-1.09	-0.11
	30	100	4.83	4.34	5.32
Size	30	300	6.61	6.11	7.10
	100	300	1.77	1.28	2.26
Type	CD	RU	6.81	6.32	7.30

Table H.13: Computation of Effects for AFL Average Transmitted Bits

		CD			RU					
Size	**RE**	**Degree 8**	**Degree 12**	**Degree 16**	**Degree 8**	**Degree 12**	**Degree 16**	**Row Sum**	**Row Mean**	**Row Effect**
30	**0.02**	173068.47	80976.19	70682.51	186646.16	89868.18	71253.08	20174837.73	112082.43	-13312.29
	0.05	181839.96	82782.74	70639.44	188891.13	93453.58	71599.39	20676187.07	114867.71	-10527.01
	0.10	188526.03	82241.79	70910.82	194840.21	92614.46	71692.31	21024768.63	116804.27	-8590.45
100	**0.02**	190034.11	87044.22	75132.58	205562.89	103067.00	82030.29	22286132.59	123811.85	-1582.87
	0.05	200657.39	89084.23	75153.27	210312.12	103295.38	81516.29	22800560.14	126669.78	1275.06
	0.10	205201.65	90502.43	75244.33	215421.95	107333.53	82777.37	23294438.07	129413.54	4018.83
300	**0.02**	201666.27	98848.15	84329.84	224997.04	119152.24	93007.21	24660022.05	137000.12	11605.40
	0.05	203059.25	95329.45	82937.19	215691.14	113600.56	88760.97	23981356.79	133229.76	7835.04
	0.10	210833.16	95930.12	82317.44	216713.45	114290.96	87952.94	24241142.08	134673.01	9278.29
Col Sum		52646588.18	24082179.31	20620422.52	55772282.86	28100276.43	21917695.85	203139445.16		
Col Mean		194987.36	89193.26	76371.94	206564.01	104075.10	81176.65		**125394.72**	
Col Effect		69592.64	-36201.46	-49022.78	81169.29	-21319.62	-44218.07			

Table H.14: Table of Effects for AFL Average Transmitted Bits

Parameter	**Mean Effect**	**Std Dev**	**Confidence Interval**	
Mean	125394.72	311.53	124882.25	125907.19
Range Error Effects				
0.02	-1096.59	440.57	-1821.33	-371.84
0.05	-472.30	440.57	-1197.05	252.44
0.10	1568.89	440.57	844.15	2293.63
Degree Effects				
8	75380.97	440.57	74656.22	76105.71
12	-28760.54	440.57	-29485.29	-28035.80
16	-46620.43	440.57	-47345.17	-45895.68
Size Effects				
30	-10809.92	440.57	-11534.66	-10085.17
100	1237.00	440.57	512.26	1961.75
300	9572.91	440.57	8848.17	10297.66
Type Effects				
CD	-5210.53	311.53	-5723.01	-4698.06
RU	5210.53	311.53	4698.06	5723.01

Table H.15: Contrast Results for AFL Average Transmitted Bits

Parameter	Level_A	Level_B	Mean Difference	Lower CL	Upper CL
	0.02	0.05	-624.28	-1349.03	100.46
RE	0.02	0.1	-2041.19	-3390.22	-1940.73
	0.05	0.1	-2665.47	-2765.94	-1316.45
	8	12	104141.51	103416.77	104866.25
Degree	8	16	122001.39	121276.65	122726.14
	12	16	17859.88	17135.14	18584.63
	30	100	-12046.92	-12771.67	-11322.18
Size	30	300	-20382.83	-21107.57	-19658.08
	100	300	-8335.91	-9060.65	-7611.16
Type	CD	RU	-10421.07	-11145.81	-9696.32

Table H.16: Computation of Effects for AFL Average Received Bits

		CD			RU					
		Degree	**Degree**	**Degree**	**Degree**	**Degree**	**Degree**	**Row**	**Row**	**Row**
Size	**RE**	**8**	**12**	**16**	**8**	**12**	**16**	**Sum**	**Mean**	**Effect**
	0.02	1269896.05	913464.48	1083715.82	1357551.85	995013.74	1089910.81	201286582.73	1118258.79	-76036.85
30	**0.05**	1325961.56	930610.90	1081732.83	1360886.20	1027861.41	1093842.82	204626871.20	1136815.95	-57479.69
	0.10	1369998.78	922421.51	1084686.69	1400030.65	1009490.23	1093955.35	206417496.40	1146763.87	-47531.78
	0.02	1398226.00	938357.03	1094976.57	1433504.66	1077326.60	1173811.43	213486068.69	1186033.71	-8261.93
100	**0.05**	1475500.92	951977.52	1096669.38	1453762.10	1060996.59	1157161.28	215882034.08	1199344.63	5048.99
	0.10	1509099.58	958860.58	1097249.36	1480432.48	1082369.34	1165573.11	218807533.02	1215597.41	21301.76
	0.02	1468991.56	1018817.89	1153295.55	1570946.80	1214467.47	1268869.06	230861649.74	1282564.72	88269.08
300	**0.05**	1470589.65	971904.73	1128761.14	1486182.98	1133013.70	1194512.22	221548932.69	1230827.40	36531.76
	0.10	1524085.56	971172.25	1116637.97	1481768.42	1126512.41	1174549.26	221841776.45	1232454.31	38158.67
Col Sum		384370489.87	257327606.65	298131759.28	390751984.10	291811544.62	312365560.48	1934758945.00		
Col Mean		1423594.41	953065.21	1104191.70	1447229.57	1080783.50	1156909.48		**1194295.65**	
Col Effect		229298.76	-241230.44	-90103.94	252933.93	-113512.15	-37386.16			

Table H.17: Table of Effects for AFL Average Received Bits

Parameter	Mean Effect	Std Dev	Confidence Interval	
Mean	1194295.65	2441.06	1190280.11	1198311.18
Range Error Effects				
0.02	1323.43	3452.17	-4355.40	7002.26
0.05	-5299.65	3452.17	-10978.48	379.18
0.10	3976.22	3452.17	-1702.61	9655.05
Degree Effects				
8	241116.34	3452.17	235437.52	246795.17
12	-177371.29	3452.17	-183050.12	-171692.46
16	-63745.05	3452.17	-69423.88	-58066.23
Size Effects				
30	-60349.44	3452.17	-66028.27	-54670.61
100	6029.61	3452.17	350.78	11708.43
300	54319.83	3452.17	48641.01	59998.66
Type Effects				
CD	-34011.87	2441.06	-38027.41	-29996.33
RU	34011.87	2441.06	29996.33	38027.41

Table H.18: Contrast Results for AFL Average Received Bits

Parameter	Level_A	Level_B	Mean Difference	Lower CL	Upper CL
	0.02	0.05	6623.08	944.25	12301.91
RE	0.02	0.1	-9275.87	-8331.61	3026.04
	0.05	0.1	-2652.79	-14954.69	-3597.04
	8	12	418487.63	412808.81	424166.46
Degree	8	16	304861.40	299182.57	310540.22
	12	16	-113626.24	-119305.07	-107947.41
	30	100	-66379.05	-72057.87	-60700.22
Size	30	300	-114669.28	-120348.10	-108990.45
	100	300	-48290.23	-53969.06	-42611.40
Type	CD	RU	-68023.74	-73702.57	-62344.92

Table H.19: Computation of Effects for AFL Average Distance Error

		CD			RU					
		Degree	**Degree**	**Degree**	**Degree**	**Degree**	**Degree**	**Row**	**Row**	**Row**
Size	**RE**	**8**	**12**	**16**	**8**	**12**	**16**	**Sum**	**Mean**	**Effect**
	0.02	1.33	0.92	0.26	2.23	0.96	0.23	177.94	0.99	-2.37
30	**0.05**	2.35	1.02	0.52	3.80	1.13	0.37	275.87	1.53	-1.82
	0.10	4.47	1.45	0.75	5.95	1.58	0.63	444.89	2.47	-0.89
	0.02	1.96	0.38	0.20	6.81	0.87	0.57	323.81	1.80	-1.56
100	**0.05**	5.24	0.91	0.49	10.15	1.99	1.05	594.66	3.30	-0.05
	0.10	10.93	1.91	0.97	14.68	4.44	1.95	1045.83	5.81	2.45
	0.02	2.49	0.65	0.37	10.21	1.81	0.68	486.22	2.70	-0.66
300	**0.05**	5.76	1.53	0.89	12.91	3.19	1.36	768.99	4.27	0.91
	0.10	12.77	3.33	1.81	17.54	5.85	2.72	1320.90	7.34	3.98
Col Sum		1418.66	363.05	187.46	2528.44	654.76	286.75	5439.11		
Col Mean		5.25	1.34	0.69	9.36	2.43	1.06		**3.36**	
Col Effect		1.90	-2.01	-2.66	6.01	-0.93	-2.30			

Table H.20: Table of Effects for AFL Average Distance Error

Parameter	**Mean Effect**	**Std Dev**	**Confidence Interval**	
Mean	3.36	0.06	3.25	3.46
Range Error Effects				
0.02	-1.53	0.09	-1.68	-1.38
0.05	-0.32	0.09	-0.47	-0.17
0.10	1.85	0.09	1.70	2.00
Degree Effects				
8	3.95	0.09	3.80	4.10
12	-1.47	0.09	-1.62	-1.32
16	-2.48	0.09	-2.63	-2.33
Size Effects				
30	-1.69	0.09	-1.84	-1.54
100	0.28	0.09	0.13	0.43
300	1.41	0.09	1.26	1.56
Type Effects				
CD	-0.93	0.06	-1.03	-0.82
RU	0.93	0.06	0.82	1.03

Table H.21: Contrast Results for AFL Average Distance Error

Parameter	Level_A	Level_B	Mean Difference	Lower CL	Upper CL
	0.02	0.05	-1.21	-1.36	-1.06
RE	0.02	0.1	-2.17	-3.53	-3.23
	0.05	0.1	-3.38	-2.32	-2.02
	8	12	5.42	5.28	5.57
Degree	8	16	6.43	6.28	6.58
	12	16	1.01	0.86	1.16
	30	100	-1.97	-2.12	-1.82
Size	30	300	-3.11	-3.26	-2.96
	100	300	-1.13	-1.28	-0.98
Type	CD	RU	-1.85	-2.00	-1.70

Table H.22: Computation of Effects for AFL Percent Localized

		CD			RU					
		Degree	**Degree**	**Degree**	**Degree**	**Degree**	**Degree**	**Row**	**Row**	**Row**
Size	**RE**	**8**	**12**	**16**	**8**	**12**	**16**	**Sum**	**Mean**	**Effect**
	0.02	100.00	100.00	100.00	100.00	100.00	100.00	18000.00	100.00	0.00
30	**0.05**	100.00	100.00	100.00	100.00	100.00	100.00	18000.00	100.00	0.00
	0.10	100.00	100.00	100.00	100.00	100.00	100.00	18000.00	100.00	0.00
	0.02	100.00	100.00	100.00	100.00	100.00	100.00	18000.00	100.00	0.00
100	**0.05**	100.00	100.00	100.00	100.00	100.00	100.00	18000.00	100.00	0.00
	0.10	100.00	100.00	100.00	100.00	100.00	100.00	18000.00	100.00	0.00
	0.02	100.00	100.00	100.00	100.00	100.00	100.00	18000.00	100.00	0.00
300	**0.05**	100.00	100.00	100.00	100.00	100.00	100.00	18000.00	100.00	0.00
	0.10	100.00	100.00	100.00	100.00	100.00	100.00	18000.00	100.00	0.00
Col Sum		27000.00	27000.00	27000.00	27000.00	27000.00	27000.00	162000.00		
Col Mean		100.00	100.00	100.00	100.00	100.00	100.00		**100.00**	
Col Effect		0.00	0.00	0.00	0.00	0.00	0.00			

(Note: Since all networks are connected and only connectivity is required for AFL to localize all nodes, AFL achieved 100% localized on all networks and all configurations.)

Table H.23: Table of Effects for AFL Percent Localized

Parameter	Mean Effect	Std Dev	Confidence Interval	
Mean	100.00	0.00	100.00	100.00
Range Error Effects				
0.02	0.00	0.00	0.00	0.00
0.05	0.00	0.00	0.00	0.00
0.10	0.00	0.00	0.00	0.00
Degree Effects				
8	0.00	0.00	0.00	0.00
12	0.00	0.00	0.00	0.00
16	0.00	0.00	0.00	0.00
Size Effects				
30	0.00	0.00	0.00	0.00
100	0.00	0.00	0.00	0.00
300	0.00	0.00	0.00	0.00
Type Effects				
CD	0.00	0.00	0.00	0.00
RU	0.00	0.00	0.00	0.00

Table H.24: Contrast Results for AFL Percent Localized

Parameter	Level_A	Level_B	Mean Difference	Lower CL	Upper CL
	0.02	0.05	0.00	0.00	0.00
RE	0.02	0.1	0.00	0.00	0.00
	0.05	0.1	0.00	0.00	0.00
	8	12	0.00	0.00	0.00
Degree	8	16	0.00	0.00	0.00
	12	16	0.00	0.00	0.00
	30	100	0.00	0.00	0.00
Size	30	300	0.00	0.00	0.00
	100	300	0.00	0.00	0.00
Type	CD	RU	0.00	0.00	0.00

Table H.25: C Matrix Used in Map Growing Regression on Average Transmitted Bits

$$\begin{bmatrix} 0.0634899 & -0.0000132 & -0.0042271 & -0.0024155 \\ -0.0000132 & 0.0000001 & 0 & 0 \\ -0.0042271 & 0 & 0.0003019 & 0 \\ -0.0024155 & 0 & 0 & 0.0048309 \end{bmatrix}$$

Table H.26: C Matrix Used in Map Growing Regression on Average Received Bits

$$\begin{bmatrix} 0.0634899 & -0.0000132 & -0.0042271 & -0.0024155 \\ -0.0000132 & 0.0000001 & 0 & 0 \\ -0.0042271 & 0 & 0.0003019 & 0 \\ -0.0024155 & 0 & 0 & 0.0048309 \end{bmatrix}$$

Table H.27: C Matrix Used in AFL Regression on ln(Average Transmitted Bits)

$$\begin{bmatrix} 0.290584 & -0.0000095 & -0.0505922 & -0.0018 & 0.0021 \\ -0.00001 & 0.0000001 & 0 & 0 & 0 \\ -0.050592 & 0 & 0.0090113 & 0 & -0.0004 \\ -0.001766 & 0 & 0 & 0.0035 & 0 \\ 0.002067 & 0 & -0.000372 & 0 & 2E-05 \end{bmatrix}$$

Table H.28: C Matrix Used in AFL Regression on ln(Average Received Bits)

$$\begin{bmatrix} 0.29577 & -0.0001257 & -0.0505687 & -0.0018 & 3E-07 & 0.0020659 \\ -0.000126 & 0.0000027 & -0.0000005 & 4E-07 & 0 & 0 \\ -0.050569 & -0.0000005 & 0.0090114 & -1E-07 & 0 & -0.000372 \\ -0.001784 & 0.0000004 & -0.0000001 & 0.0035 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0.002066 & 0 & -0.000372 & 0 & 0 & 0.0000155 \end{bmatrix}$$

Appendix I. MICA2DOT Calculations

Table I.1: MICA2DOT Data Summary [Cro05]

Processor Full Operation	8mA (4Mhz)
Radio, Receive	10mA
Radio, Transmit (full power)	27mA
Data Rate	38.4 Kbaud
Typical Battery	Coin (Li-ion)
Typical Battery Capacity	560mA-hr
Operating Voltage Range	3.6 to 2.7 Volts

Map Growing Energy Consumption Model for the MICA2DOT

$$\begin{aligned} E &= (-20123 - 0.574S + 3408D - 538T) \times 1.95313\text{E-}07 \frac{\text{mA-hr}}{\text{bit}} \\ &\quad + (-697537 - 62.7S + 82394D - 23960T) \times 7.2338\text{E-}08 \frac{\text{mA-hr}}{\text{bit}} \\ &\quad + 0.074621628\text{mA-hr} \end{aligned} \tag{I.1}$$

Table I.2: Example Map Growing Energy Consumption Model Results

S	**D**	**T**	**Predicted ATB**	**Predicted ARB**	**mA-hr Consumed**	**% of Batt. Consumed**
400	14	CD	26823.73	406962.4	0.109299469	0.019517762
200	10	RU	13843.48	113875.6	0.085562962	0.015279100
1000	18	CD	39453.19	698948.4	0.132887833	0.023729970
Average Map Growing Response			21225.04	317935.8	0.101765971	0.018172495

AFL Energy Consumption Model for the MICA2DOT

$$\begin{aligned} E &= e^{(15.4+0.000647S-0.547D-0.0786T+0.0179D^2)} \times 1.95313\text{E-}07 \frac{\text{mA-hr}}{\text{bit}} \\ &\quad + e^{(16.3+0.00108S-0.397D-0.0539T-2\text{E-}06S^2+0.0153D^2)} \times 7.2338\text{E-}08 \frac{\text{mA-hr}}{\text{bit}} \\ &\quad + 0.001275094\text{mA-hr} \end{aligned} \tag{I.2}$$

Table I.3: Example AFL Energy Consumption Model Results

S	D	T	Predicted ATB	Predicted ARB	mA-hr Consumed	% of Batt. Consumed
400	14	CD	90653.51582	984904.5383	0.090226847	0.016111937
200	10	RU	137805.6999	1211536.565	0.115830357	0.020683992
1000	18	CD	148197.3336	475092.1195	0.064587082	0.011533407
Average AFL Response			125394.72	1194295.65	0.112159165	0.020028422

Worst Case Map Growing Energy Consumption Model for the MICA2DOT

$$\begin{aligned} E &= (-20123 - 0.574S + 3408D - 538T) \times 1.95313\text{E-}07 \frac{\text{mA-hr}}{\text{bit}} \\ &\quad +(-0.05 + 6.8333\text{E-}3S)\text{mA-hr} \\ &\quad +0.074621628\text{mA-hr} \end{aligned} \tag{I.3}$$

Table I.4: Example Worst Case Map Growing Energy Consumption Model Results

Size	Degree	Type	Predicted ATB	Approx Run Time(min)	mA-hr Consumed	% of Batt. Consumed
400	14	CD	26823.73	16.1	2.763193971	0.493427495
200	10	RU	13843.48	7.9	1.3939921	0.248927161
1000	18	CD	39453.19	40.7	6.865660663	1.226010833
Average Map Growing Response			21225.04	5.563	1.005933811	0.179631038

Worst Case AFL Energy Consumption Model for the MICA2DOT

$$\begin{aligned} E &= e^{(15.4+0.000647S-0.547D-0.0786T+0.0179D^2)} \times 1.95313\text{E-}07 \frac{\text{mA-hr}}{\text{bit}} \\ &\quad +(2 + 6.6667\text{E-}3S)\text{mA-hr} \\ &\quad +0.001275094\text{mA-hr} \end{aligned} \tag{I.4}$$

Table I.5: Example Worst Case AFL Energy Consumption Model Results

Size	Degree	Type	Predicted ATB	Approx. Run Time(min)	mA-hr Consumed	% of Batt. Consumed
400	14	CD	90653.51582	28	4.685647526	0.836722772
200	10	RU	137805.6999	20	3.361523603	0.600272072
1000	18	CD	148197.3336	52	8.696886552	1.553015456
Average AFL Response			125394.72	17.72	2.979099584	0.531982069

Bibliography

APL99. K. Amouris, S. Papavassiliou, and M. Li. A position-based multi-zone routing protocol for wide area mobile ad-hoc networks. In *Proceedings of the Vehicular Technology Conference (VTC '99)*, Houston, TX, May 16-20 1999.

ASSC02. I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *Computer Networks (Elsevier) Journal*, 38(4):393–422, March 2002.

AV04. C. Alippi and G. Vanini. Wireless sensor networks and radio localization: A metrological analysis of the MICA2 received signal strength indicator. In *In Proceedings 29th IEEE Local Computer Networks*. Institute of Electrical and Electronics Engineers (IEEE), 2004.

BHE01. N. Bulusu, J. Heidemann, and D. Estrin. Adaptive beacon placement. In *Proceedings of the 21st Annual International Conference on Distributed Computing Systems (ICDCS-21)*, volume 2, April 2001.

BHET04. N Bulusu, J. Heidemann, D. Estrin, and Tran T. Self-configuring localization systems: Design and experimental evaluation. *ACM Transactions on Embedded Computing Systems*, 3:24–60, 2004.

BP00. P. Bahl and V. N. Padmanabhan. Radar: An inbuilding RF-based user location and tracking system. In *Proceedings of the Nineteeth Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Infocom 2000)*, volume 2, March 26-30 2000.

CHH01. S. Capkun, M. Hamdi, and J.P. Hubaux. GPS-free positioning in mobile ad-hoc networks. In *Proceedings of the 34th Hawaiian International Conference on System Sciences (HICCSSS'01)*, Maui, Hawaii, 2001.

Cro05. Crossbow Technology Incorporated. http://www.xbow.com/, July 2005.

CYE+03. J. Chen, L. Yip, J. Elson, H. Wang, D. Maniezzo, R.E. Hudson, K. Yao, and D. Estrin. Coherent acoustic array processing and localization on wireless sensor networks. In *Proceedings of the IEEE*, volume 91, pages 1154–1162, August 2003.

EBD+02. L. Evers, W. Bach, D. Dam, M. Jonker, J. Scholten, and P. J. M. Havinga. An iterative Quality-Based localization algorithm for ad hoc networks. In *1st Int. Conf. on Pervasive Computing (Pervasive)*, pages 55–61, Zurich, Switzerland, Aug 2002.

Enc97. Encoder: The Newsletter of the Seattle Robotics Society. Ultrasonics and Robotics. http://www.seattlerobotics.org/encoder/may97/sonar2.html, 1997.

GE01. L. Girod and D. Estrin. Robust range estimation for localization in ad hoc sensor networks. http://lecs.cs.ucla.edu/girod/papers/NLOS.ps, 2001.

GKW+00. D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker. Complex behavior at scale: An experimental study of low-power wireless sensor networks. Technical Report UCLA Technical Report UCLA/CSD-TR 02-0013, University of California at Los Angeles, University of California at Los Angeles, Los Angeles, CA, 2000.

HHB+03. T. He, C. Huang, B. Blum, J. Stankovic, and T. Abdelzaher. Range-free localization schemes for large scale sensor networks. In *Proceedings of the 9th ACM International Conference on Mobile Computing and Networking (MobiCom '03)*, pages 81–95, San Diego, CA, September 2003.

HWB00. J. Hightower, R. Want, and G. Borriello. Spoton: An indoor 3D location sensing technology based on RF signal strength. Technical Report UW CSE Tech. Rep. 00-02-02, University of Washington, University of Washington, Department of Computer Science and Engineering, Seattle, WA, February 2000.

IS03. Rajagopal Iyengar and Biplab Sikdar. Scalable and distributed GPS free positioning for sensor networks. website, Rensselaer Polytechnic Institute, Troy, NY, 2003.

Jai91. Raj Jain. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurment, Simulation, and Modeling.* Wiley-Interscience, New York, NY, May 1991.

LH03. Dan Li and Yu Hen Hu. Energy-based collaborative source localizatn using acoustic microsensor array. In *EURASIP Journal on Applied Signal Processing*, pages 321–337, March 2003.

LJD+00. J. Li, J. Jannotti, D. De Couto, D. Karger, and R. Morris. A scalable location service for geographic ad-hoc routing. In *Proceedings of the 6th ACM International Conference on Mobile Computing and Networking (MobiCom '00)*, pages 120–130, August 2000.

LR03. Koen Langendoen and Niels Reijers. Distributed localization in wireless sensor networks: a quantitative comparison. In *Elsevier Computer Networks 43*, pages 499–518, Netherlands, 2003.

LSS04a. Xiaoli Li, Hongchi Shi, and Yi Shang. A map-growing localization algorithm for ad-hoc wireless sensor networks. In *Proceedings of 10th International Conference on Parallel and Distributed Systems(ICPADS-10)*, pages 395–402, July 2004.

LSS04b. Xiaoli Li, Hongchi Shi, and Yi Shang. A partial-range-aware localization algorithm for ad hoc wireles sensor networks. In *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*, pages 77–83, November 2004.

LW05. K. Lorincz and M. Welsh. Motetrack: A robust, decentralized location tracking system for disaster response. http://www.eecs.harvard.edu/ konrad/projects/motetrack/, 2005.

MGZN03. Jian Ma, Min Gao, Yanmin Zhu, and Lionel M. Ni. Quality based anchor-free localization with refinement in sensor networks. http://ihome.ust.hk/ zhuym/research/publications/conference/Localization_submission_version.pdf, 2003.

Nag99. R. Nagpal. Organizing a global coordinate system from local information on an amorphous computer. website, Massachusetts Institute of Technology, Boston, MA, 1999.

NLLP03. L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil. Landmarc: Indoor location sensing using active RFID. In *Proceedings of IEEE PerCom 2003*, Dallas, TX, March 2003.

NN01. D. Niculescu and B. Nath. Ad hoc positioning system (APS). In *Proceedings of GLOBECOM*, San Antonio, TX, November 2001.

NN03. D. Niculescu and B. Nath. Ad hoc positioning system (aps) using aoa. In *Proceedings of the Twenty Second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Infocom 2003)IEEE INFOCOM*, pages 2037–2040, Salt Lake City, UT, April 2003.

PBDT03. Nissanka B. Priyantha, Hari Balakrishnan, Erik Demaine, and Seth Teller. *Anchor Free Distributed Localization in Sensor Networks*. MIT Press, Boston, MA, 2003.

PCB00. N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proceedings of the 6th ACM International Conference on Mobile Computing and Networking (MobiCom '00)*, Boston, MA, August 2000. ACM.

Sav04. A. Savvides. Location discovery part II networked embedded systems and sensor networks. http://www.eng.yale.edu/enalab/courses/eeng460a/lec05.ppt, 2004.

SH03. X. Sheng and Y. Hu. Energy based acoustic source localization. In *Proceedings of Second Annual Information Processing in Sensor Networks*, pages 285–300. IEEE Aerospace and Electronic Systems Society, April 2003.

SHS01. A. Savvides, C. Han, and M. Srivasta. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proceedings of 7th ACM International Conference on Mobile Computing and Networking (MobiCom '01)*, pages 166–179, July 2001.

SHS04. R. Stoleru, T. He, and J. A. Stankovic. Walking GPS: A practical solution for localization in manually deployed wireless sensor networks. In *1st IEEE*

Workshop on Embedded Networked Sensors (EmNetS), Tampa, Florida, 2004.

SPS02. A. Savvides, H. Park, and M. Srivastava. The bits and flops of the n-hop multilateration primitive for node localization problems. In *In First ACM International Workshop on Wireless Sensor Networks and Application*, Atlanta, GA, September 2002.

SRB01. C. Savarese, J. Rabaey, and J. Beutel. Locationing in distributed ad-hoc wireless sensor networks. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '01)*, pages 2037–2040, Salt Lake City, UT, May 2001.

SRL02. C. Savarese, J. Rabaey, and K. Langendoen. Robust positioning algorithms for distributed ad-hoc wireless sensor networks. In *Proceedings of USENIX Annual Technical Conference, General Track*, pages 317–327, Monterey, CA, June 2002.

SRZF03. Yi Shang, Wheeler Ruml, Ying Zhang, and Marcus Fromherz. Localization from mere connectivity. In *Proceedings of Fourth Annual ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2003)*, pages 201–212. ACM, June 2003.

TGB+04. M. Terwilliger, A. Gupta, V. Bhuse, Z. H. Kamal, and M.A. Salahuddin. A localization system using wireless network sensors: A comparison of two techniques. In *In Proc. of the First Workshop on Positioning, Navigation and Communication*, Hannover, Germany, March 2004.

TP03. Ankur Tarnacha and Thomas F. La Porta. E-strobe: An adaptive beacon activation algorithm for sensor localization. In *Proceedings of IEEE Vehicular Technology Conference (VTC) 2003 Symposium on Wireless Ad hoc, Sensor, and Wearable Networks (VTC '03)*, Orlando, Florida, October 2003. IEEE.

Tri05. Trimble. All about gps. http://www.trimble.com/gps/triangulating3.html, May 2005.

WC02. Kamin Whitehouse and David E. Culler. Calibration as parameter estimation in sensor networks. In *WSNA*, pages 59–67, 2002.

Wei05. Eric W. Weisstein. Relative error. From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/RelativeError.html, 2005.

WG97. Paul R. Wolf and Charles D. Ghilani. *Adjustment Computations: Statics and Least Squares in Surveying and GIS*. Wiley-Interscience, New York, NY, 1997.

ZG03. Y. J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor network. In *1st ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Los Angeles, CA, 2003.

ZHS04. G. Zhou, T. He, and J. A. Stankovic. Impact of radio asymmetry on wireless sensor networks. In *2nd International Conference on Mobile Systems, Applications, and Services (Mobisys)*, Boston, MA, 2004.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
23–03–2006	Master's Thesis	Aug 2004 — Mar 2006

4. TITLE AND SUBTITLE

Evaluation of Energy Costs
and Error Performance of
Range-Aware, Anchor-Free Localization Algorithms
for Wireless Sensor Networks

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Gustav Julio Jordt, Capt, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way Bldg 640
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GCE/ENG/06-02

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Mr. William Koenig (William.Koenig@wpafb.af.mil)
AFRL/IFSC (AFMC)
2241 Avionics Circle
WPAFB, OH 45433
DSN785-4709x3172

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approval for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT This research examines energy and error tradeoffs in Anchor-Free Range-Aware Wireless Sensor Network (WSN) Localization algorithms. A concurrent and an incremental algorithm (Anchor Free Localization (AFL) and Map Growing) are examined under varying network sizes, densities, deployments, and range errors. Despite current expectations, even the most expensive configurations do not expend significant battery life (at most 0.4%), implying little energy can be conserved during localization. Due to refinement, AFL is twice as accurate, using 6 times the communication. For both, node degree affects communication most. As degree increases, Map Growing communication increases, while AFL transmissions drop. Nodes with more neighbors refine quicker with fewer messages. At high degree, many nodes receive the same message, overpowering the previous effect, and raising AFL received bits. Built from simulation data, the Energy Consumption Model predicts energy usage of incremental and concurrent algorithms used in networks with varying size, density, and deployments. It is applied to current wireless sensor nodes. Military WSNs should be flexible, cheap, and long lasting. Anchor-Free, Range-Aware algorithms best fit this need.

15. SUBJECT TERMS

Wireless Sensor Networks (WSN), wireless sensor nodes, ad-hoc, localization, autolocalization, range-aware, anchor-free, distributed algorithms, energy conservation, power conservation

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Rusty O. Baldwin
U	U	U	UU	180	19b. TELEPHONE NUMBER *(include area code)* (937) 255–6565, ext 4445